

## Simulation Techniques for Intense Beams

### Outline

#### 1. Why Numerical Simulation

#### 2. Classes of Intense Beam Simulations

A. Particle Methods

B. Distribution Methods

C. Moment Methods.

#### 3. Numerical Methods

A. Discretizations

- Derivatives

- Integrals

B. Application to Moment Methods

- Euler and Runge-kutta advances

- Example Application

#### 4. Numerical Methods for Particle and Distribution Methods

A. Field discretizations

B. Particle Methods and "PIC" Codes for solving  
Vlasov's equation.

- Leapfrog advance

- Field solution

- Particle weighting

- Advance cycle

- Initialization

- Numerical convergence

- Examples

### C. Distribution Methods

- Similarities with particle methods

will not

Cover  
class  
mention

in

only

qualitatively

{ - Distribution advance

{ - Examples

### 5. Overview of the WARP code

References.

# Simulation Techniques for Intense Beams

## §1 Overview: Why Numerical Simulation?

- Builds intuition on intense beam dynamics.

The primary purpose of simulation is to gain insight on trends, sensitivities, etc. - not specific numbers!

In contrast to laboratory experiments:

- Full nonintrusive beam diagnostics are possible
- Effects can be turned on and off.

This insight can also be applied to motivate approximations in analytical analysis.

- Allows analysis of more realistic situations that can be intractable in analytical analysis.

- Realistic geometries
- Non-ideal distributions
- Combined effects
- Large amplitude (nonlinear) effects.

- Expected performance of specific machines can be quantified.

- Computers and numerical methods / libraries are becoming more powerful enabling both analysis of more realistic problems and better numerical convergence.

- Processor speed increasing
- Parallel machine architectures
- Memory larger
- Numerical methods improved
- Libraries of debugged code modules

} Hardware  
bigger and faster

} Software  
more developed

Although simulations are increasingly powerful and valuable in the analysis of intense beams, simulations should not be used to exclusion:

- Parametric scaling is very important in machine design. Often it is hardest to understand what specific choices should be made in physical aperture sizes, etc. Although scaling can be explored with simulation, Analytical analysis often best illustrates the trade offs, sensitivities, and relevant combinations of parameters.
- Concepts often fail due to limits of technology (fabrication tolerances, material failures, unanticipated properties, ...) and hence full laboratory testing is vital. Many understood classes of error can be probed with simulation though - unanticipated sources are most dangerous.
- Economic realities often severely limit what can be constructed and such limitations must be understood so you do not simulate something unobtainable.

The highest understanding and confidence is achieved when results from analytic theory, numerical simulation, and experiment all converge.

These days numerical simulation skills also make you highly employable in many areas of accelerator and beam physics!

Numerous programming languages are employed in numerical simulations of intense beams.

The most common today are:

Fortran, Fortran 90

C, C++

Java

Results are analyzed with a variety of graphics packages:

NCAR, gists, gnuplot, idl, narcisse, ....

Perhaps the most modern and flexible way simulation packages can be structured is to link routines programmed in fast, compiled code with an interactive interpreter such as:

python, basis, yorick

- Allows routines to be coded in mixed languages and (for python) in modern, object-oriented structure.
- Flexible reconfiguration of code modules
  - possible to adapt to specific (unanticipated) needs
    - reduces need for recompilation and cumbersome structures for special uses.
    - aids cross-checking problems and debugging
  - "Steering" of code during runs to address unanticipated effects.

Discussing particular programming languages and graphics packages is beyond the scope of this class. Here our goal is to survey numerical simulation methods employed without presenting details of specific implementations.

However, — We will show examples based on the "WARP" particle-in-cell code developed for intense beam simulation at LLNL and LBNL.

WARP - Named since it works on a "warped" Cartesian mesh with bends.

WARP is a family of particle-in-cell code tools built around a common python interpreter for flexible operation. It is optimized for the simulation of intense beams with self-consistent electrostatic space-charge forces.

More on WARP later after discussion of methods etc.

## 3 Classes of Intense Beam Simulations

There are 3 distinct classes of modeling of intense ion beams applicable to numerical simulation

1/ Particle Methods

2/ Distribution Methods

3/ Moment Methods

All of these draw heavily on methods developed for the simulation of neutral plasmas.

The main differences:

- Lack of overall charge neutrality
  - Single species typical (though electron + ion simulations are common too).
- Directed motion of the beam.
- Applied field descriptions are setup for beam optical elements and accelerating structures.

Review / contrast these methods before discussing specific numerical implementations.

## 1 Particle Methods

Classical point particles are advanced with self-consistent interactions given by the Maxwell Equations.

- Most general: If actual number of particles are used, this is approx. the physical beam.

Equations of motion (time domain, 3d, for generality)

i<sup>th</sup> particle:

$$\frac{d\vec{p}_i}{dt} = \vec{F}_i = g(\vec{E} + \frac{d\vec{x}_i}{dt} \times \vec{B})$$

$$m\gamma_i \frac{d\vec{x}_i}{dt} = \vec{p}_i; \quad \gamma_i = [1 + \vec{p}_i^2/mc^2]^{1/2}$$

Initial conditions:

$$\{\vec{x}_i(t=0), \vec{p}_i(t=0)\}$$

particle orbits

$\vec{x}_i(t), \vec{p}_i(t)$  solved as a function of time.

Fields (Electromagnetic in most general form)

$\nabla \cdot \vec{E} = \frac{p}{\epsilon_0}$	charge density $p(\vec{x}, t) = p_{ext}(\vec{x}, t) + \sum_i g S \delta(\vec{x} - \vec{x}_i(t))$
$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t}$	external (applied) $\vec{J}(\vec{x}, t) = \vec{J}_{ext}(\vec{x}, t) + \sum_i g \frac{d\vec{x}_i}{dt} S [\vec{x} - \vec{x}_i(t)]$
$\nabla \times \vec{B} = \mu_0 \vec{J} + \mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t}$	particle beam
$\nabla \cdot \vec{B} = 0$	
+ Boundary Conditions on $\vec{E}, \vec{B}$	

## 2/ Distribution Methods:

- Based on reduced (statistical) <sup>continuum</sup> models of the beam.
- Two classes (microscopic) kinetic models and (macroscopic) fluid models.
- Here distribution means a function of cont. variables <sup>e.g.  $f(\vec{x}, \vec{p}, t)$</sup>  in Vlasov model.

### 2a Kinetic Model (Vlasov, neglect collisions)

- Obtained from statistical averages of particle formulation

#### Equations of Motion

$$\left\{ \frac{d}{dt} + \vec{v} \cdot \frac{d}{d\vec{x}} + g(\vec{E} + \vec{v} \times \vec{B}) \right\} f(\vec{x}, \vec{p}, t) = 0$$

$$\vec{v} = \frac{\vec{p}}{m} = \frac{\vec{p}/m}{(1 + \vec{p}^2/m^2 c^2)^{1/2}}$$

initial conditions:

$$f(\vec{x}, \vec{p}, t=0)$$

Distribution evolution

$f(\vec{x}, \vec{p}, t)$  evolved from  $t=0$ .

$\vec{x}, \vec{p}, t$  indep. variables.

#### Fields

Same as particle case with different expressions of  $\rho, \vec{j}$ :

$$\nabla \cdot \vec{E} = \frac{\rho}{\epsilon_0}$$

$$f(\vec{x}, t) = f_{ext}(\vec{x}, t) + g \int d^3 p f(\vec{x}, \vec{p}, t)$$

$$\nabla \times \vec{E} = - \frac{\partial \vec{B}}{\partial t}$$

external (applied)  
beam

$$\nabla \times \vec{B} = \mu_0 \vec{j} + \mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t}$$

$$\vec{j}(\vec{x}, t) = \vec{j}_{ext}(\vec{x}, t) + g \int d^3 p \vec{v} f(\vec{x}, \vec{p}, t)$$

$$\nabla \cdot \vec{B} = 0$$

+ Boundary conditions on  
 $\vec{E}, \vec{B}$

The Vlasov equation is essentially a continuity equation for an incompressible "fluid"  $f$  in 6D phase-space. To see this, note that:

$$\frac{\partial}{\partial \vec{p}} \cdot \vec{v} \times \vec{B} = 0$$

so the Vlasov equation can be expressed as:

$$\frac{\partial f}{\partial t} + \frac{\partial}{\partial \vec{x}} \cdot (\vec{v} f) + \frac{\partial}{\partial \vec{p}} \cdot \left( g [\vec{E} + \vec{v} \times \vec{B}] f \right) = 0$$

which is manifestly the form of a continuity equation in 6D phase-space.  $\Rightarrow$  probability is not created or destroyed.

The effect of collisions can be added to this formulation by adding a collision operator:

$$\frac{\partial f}{\partial t} + \vec{v} \cdot \frac{\partial f}{\partial \vec{x}} + g [\vec{E} + \vec{v} \times \vec{B}] \cdot \frac{\partial f}{\partial \vec{p}} = \frac{\partial f}{\partial t}_{\text{coll}}$$

For most applications in beam physics

$$\frac{\partial f}{\partial t}_{\text{coll}}$$

can be neglected (see Intro. lectures). For exceptional cases specific forms of collision terms can be found in Nicholson, "Intro to Plasma Theory," Wiley 1983.

We will find later that the common Particle-in-Cell (PIC) is not really a particle method but rather a method that uses a collection of smoothed "macro" particles to simulate Vlasov's equation. This roughly can be understood by noting that Vlasov's equation can be interpreted as

$$\frac{d}{dt} f(\vec{x}, \vec{p}, t) = 0$$

} total derivative along a test particle.

$\Rightarrow$  Advance test particles in a continuous field "fluid" to erase collisions.

But Remember:

PIC is a method to solve Vlasov's equation <u>not</u> a discrete particle method.
---

This notion should become clear after these lectures.

## 2b. Fluid Model

- Obtained from further averages of kinetic models.
- Described in terms of "macroscopic" variables (density, flow velocity, pressure, ...)
- Models must be closed (truncated) at some order via physically motivated assumptions (cold or negligible heat flow, ...)

Density  $n$ : 
$$n(\vec{x}, t) = \int d^3 p f(\vec{x}, \vec{p}, t)$$

Flow velocity  $\vec{V}$ : 
$$n \vec{V}(\vec{x}, t) = \int d^3 p \vec{v} f(\vec{x}, \vec{p}, t)$$

$\vec{P}$  
$$n \vec{P}(\vec{x}, t) = \int d^3 p \vec{p} f(\vec{x}, \vec{p}, t)$$

Pressure Tensor  $\rho_{ij}$ : 
$$n \rho_{ij}(\vec{x}, t) = \int d^3 p [p_i - P_i(\vec{x}, t)] [\delta_j - V_j(\vec{x}, t)] f(\vec{x}, \vec{p}, t)$$

Higher Rank  
Objects

## Equations of Motion (Eulerian approach)

continuity: 
$$\frac{\partial n}{\partial t} + \frac{\partial}{\partial \vec{x}} \cdot [n \vec{V}] = 0$$

Force: 
$$n \left( \frac{\partial}{\partial t} + \vec{V} \cdot \frac{\partial}{\partial \vec{x}} \right) \vec{P}_i + \sum_j \frac{\partial}{\partial x_j} P_{ij} = n g [\vec{E} + \vec{V} \times \vec{B}]$$

Higher order - Models must be truncated.

Simplist: cold fluid ( $P_{ij} \equiv 0$ )

## Field

Same as the kinetic and particle case with

$$\vec{P}(\vec{x}, t) = \vec{P}_{ext}(\vec{x}, t) + g n(\vec{x}, t)$$

$$\vec{J}(\vec{x}, t) = \vec{J}_{ext}(\vec{x}, t) + g n(\vec{x}, t) \vec{V}(\vec{x}, t)$$

## Aside: Lagrangian Formulations of Distribution Methods

In certain kinetic and especially fluid models it can be convenient to adopt so-called "Lagrangian" models.

Eulerian fluid model: flow quantities specified in terms of space ( $\vec{x}$ ) and time ( $t$ ).  
 $\vec{V} = \vec{V}(\vec{x}, t)$ .

Lagrangian fluid model:

Identify parts of the flow with objects (material element) and follow the flow in time ( $t$ ).

(shape of elements must generally change in  $t$ )

In reality many particle methods are really hybrid Lagrangian too!

- Smoothing methods in the common PIC (Particle In Cell) method (to be covered) typically employ shaped particles that may be thought of as Lagrangian elements representing a Vlasov fluid.

Example: 1D Lagrangian model of the longitudinal evolution of a cold beam.

Coordinates	$z_0$	$z_1$	$z_2$	$z_3$	$z_4$	.....	$z_N$
Charges	$Q_{1/2}$	$Q_{3/2}$	$Q_{5/2}$	$Q_{7/2}$			
Masses	$m_{1/2}$	$m_{3/2}$	$m_{5/2}$	$m_{7/2}$			
Velocities	$v_0$	$v_1$	$v_2$	$v_3$	$v_4$	.....	$v_N$

$$z_i \text{ slice boundaries} \quad Q_{i+1/2} \text{ Fixed} \quad \frac{Q_{i+1/2}}{m_i + v_i} = g \\ v_i \text{ velocities of slice boundaries.} \quad m_{i+1/2} \text{ Fixed} \quad \frac{m_{i+1/2}}{m_i + v_i} = \text{const}$$

Solve the equations of motion:

$$\frac{dz_i(t)}{dt} = v_p(t)$$

for single species.  
(set initial cond.)

$$\frac{dv_i(t)}{dt} = \left( \frac{g}{m} \right) E_z(z_i, t)$$

for all the slice boundaries. Several methods might be used to calculate  $E_z$ :

1) Take "slices" to have some radial extent modeled by a  $\perp$  envelope etc. and deposit the  $Q_{i+1/2}$  onto a grid and solve:

$$\nabla^2 \phi = -\rho \quad E_z = -\frac{\partial \phi}{\partial z}$$

subject to  $E_r \rightarrow 0$  as  $|z| \rightarrow \infty$

2) Employ a "g"-factor model

$$E_z = \frac{-g}{4\pi\epsilon_0} \frac{\partial \lambda}{\partial z}$$

$\lambda$  calculated from  $Q_i$  and radial extent of elements etc.

3) Pure-1d model and use Gauss' Law

### 3/ Moment Methods (with all phase space variations integrated over)

- Most reduced description of intense beam.
- Beam represented by a finite (closed and truncated) set of moments that are advanced from initial values.  
Here by moments, we mean functions of a single variable's sort.
- Such models usually are not fully self-consistent but are commonly used for lattice design.
  - Some special cases such as a stable L KV equilibrium distribution are consistent with truncated moment description (rms envelope egn.)
  - Typically derived from assumed distributions with self-similar evolution,

Many moment models exist. Illustrate with examples for L beam evolution:

1st order moments

$$\langle \dots \rangle_1 = \frac{\int dx dy f dx' dy' \dots f}{\int dx dy f dx' dy' f}$$

$$\bar{x}_c = \langle \bar{x} \rangle_1$$

centroid coordinate

$$\bar{x}'_c = \langle \bar{x}' \rangle_1$$

centroid angle

$$\Delta = \langle \frac{\delta p}{p} \rangle_1 = \langle \delta \rangle_1$$

off momentum

:

:

2nd order moments

x moments	y moments	x-y cross moments	dispersive moments
$\langle x^2 \rangle_1$	$\langle y^2 \rangle_1$	$\langle xy \rangle_1$	$\langle x\delta \rangle_1, \langle y\delta \rangle_1$
$\langle xx' \rangle_1$	$\langle yy' \rangle_1$	$\langle x'y \rangle_1, \langle xy' \rangle_1$	$\langle x'\delta \rangle_1, \langle y'\delta \rangle_1$
$\langle x'^2 \rangle_1$	$\langle y'^2 \rangle_1$	$\langle x'y' \rangle_1$	$\langle \delta^2 \rangle_1$

Note:

Typically convenient to subtract centroid from higher order moments

$$\tilde{x} \equiv x - x_c \quad , \quad \tilde{x}' \equiv x' - x'_c$$

$$\tilde{y} \equiv y - y_c \quad , \quad \tilde{y}' \equiv y' - y'_c$$

$$\langle \tilde{x}^2 \rangle_1 = \langle (x - x_c)^2 \rangle_1 = \langle x^2 \rangle_1 - x_c^2, \text{ etc.}$$

3rd order moments

Analogous.  $\langle x^3 \rangle_1, \langle x^2 y \rangle_1, \dots$

Many quantities of physical interest are expressed in terms of moments:

$$\text{Statistical } \left\{ \begin{array}{l} \bar{x} = 2\langle \tilde{x}^2 \rangle_1^{1/2} \end{array} \right.$$

$$\text{Beam size: } \left\{ \begin{array}{l} \bar{y} = 2\langle \tilde{y}^2 \rangle_1^{1/2} \end{array} \right.$$

$$\text{Statistical } \left\{ \begin{array}{l} \bar{E}_x = 4[\langle \tilde{x}^2 \rangle_1 \langle \tilde{x}'^2 \rangle_1 - \langle \tilde{x} \tilde{x}' \rangle_1^2]^{1/2} \end{array} \right.$$

$$\text{Emittances } \left\{ \begin{array}{l} \bar{E}_y = 4[\langle \tilde{y}^2 \rangle_1 \langle \tilde{y}'^2 \rangle_1 - \langle \tilde{y} \tilde{y}' \rangle_1^2]^{1/2} \end{array} \right.$$

### Equations of Motion

- Can be expressed in terms of moments or combinations of moments that are of physical interest.
- Moments are advanced from specified initial values.

Ex:  $\delta = 0$ , 2nd order truncation, zero x-y cross moments  
linear focusing lattice, elliptical beam with uniform charge.

$$\frac{d\bar{x}_c}{ds} = -R_x(s)\bar{x}_c + \text{image terms} ; \quad \frac{d\bar{x}_c}{ds} = \bar{x}'$$

$$\frac{d\bar{y}_c}{ds} = -R_y(s)\bar{y}_c + \text{image terms} ; \quad \frac{d\bar{y}_c}{ds} = \bar{y}'$$

$\langle \tilde{x}^2 \rangle_1$	$2\langle \tilde{x} \tilde{x}' \rangle_1$
$\langle \tilde{x} \tilde{x}' \rangle_1$	$\langle \tilde{x}'^2 \rangle_1 - R_x(s)\langle \tilde{x}^2 \rangle_1 + Q\langle \tilde{x}^2 \rangle_1^{1/2}/[4\langle \tilde{x}^2 \rangle_1^{1/2}(\langle \tilde{x}^2 \rangle_1^{1/2} + \langle \tilde{y}^2 \rangle_1^{1/2})]$
$\frac{d}{ds}\langle \tilde{x}^2 \rangle_1$	$-2R_x(s)\langle \tilde{x} \tilde{x}' \rangle_1 + 2Q\langle \tilde{x} \tilde{x}' \rangle_1/[4\langle \tilde{x}^2 \rangle_1^{1/2}(\langle \tilde{x}^2 \rangle_1^{1/2} + \langle \tilde{y}^2 \rangle_1^{1/2})]$
$\langle \tilde{y}^2 \rangle_1$	$2\langle \tilde{y} \tilde{y}' \rangle_1$
$\langle \tilde{y} \tilde{y}' \rangle_1$	$\langle \tilde{y}'^2 \rangle_1 - R_y(s)\langle \tilde{y}^2 \rangle_1 + Q\langle \tilde{y}^2 \rangle_1^{1/2}/[4\langle \tilde{y}^2 \rangle_1^{1/2}(\langle \tilde{x}^2 \rangle_1^{1/2} + \langle \tilde{y}^2 \rangle_1^{1/2})]$
$\langle \tilde{y}'^2 \rangle_1$	$-2R_y(s)\langle \tilde{y} \tilde{y}' \rangle_1 + 2Q\langle \tilde{y} \tilde{y}' \rangle_1/[4\langle \tilde{y}^2 \rangle_1^{1/2}(\langle \tilde{x}^2 \rangle_1^{1/2} + \langle \tilde{y}^2 \rangle_1^{1/2})]$

Or using  $\frac{d}{ds}\bar{E}_x^2 = 0 = \frac{d}{ds}\bar{E}_y^2$ , the 2nd order moment equations can be equivalently expressed as:

$$\frac{d\bar{r}_x'}{ds} + R_x\bar{r}_x - \frac{Q}{\bar{r}_x + \bar{r}_y} - \frac{\bar{E}_x^2}{\bar{r}_x^3} = 0 ; \quad \frac{d\bar{r}_x}{ds} = \bar{r}_x'$$

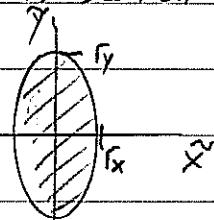
$$\frac{d\bar{r}_y'}{ds} + R_y\bar{r}_y - \frac{Q}{\bar{r}_x + \bar{r}_y} - \frac{\bar{E}_y^2}{\bar{r}_y^3} = 0 ; \quad \frac{d\bar{r}_y}{ds} = \bar{r}_y'$$

Field

- In moment methods fields are typically calculated using averages over assumed distribution forms.  
- often analytic solutions employed.

Example

Envelope model employed above assumes an elliptical cross-section beam with uniform charge in free space.



$$\text{For: } \tilde{x}^2/r_x^2 + \tilde{y}^2/r_y^2 \leq 1$$

$\lambda$  = line charge density

$$E_{\tilde{x}} = \frac{\lambda \tilde{x}}{\pi \epsilon_0 r_x (r_x + r_y)}, \quad E_{\tilde{y}} = \frac{\lambda \tilde{y}}{\pi \epsilon_0 r_y (r_x + r_y)}$$

these forms are employed in the derivation of the moment equations of motion presented (KV Envelope Equation).

Moment models generally form an infinite chain of equations that do not truncate.

To be useful the moments must be truncated.

Truncations are usually carried out via field model assumptions. For example, the KV model field assumption above leads to a truncation of moments at 2nd order with

$$E_x^2 = 16 [\langle x^2 \rangle_2 \langle x'^2 \rangle_1 - \langle xx' \rangle_1^2] = \text{const.}$$

$$E_y^2 = 16 [\langle y^2 \rangle_2 \langle y'^2 \rangle_1 - \langle yy' \rangle_1^2] = \text{const.}$$

Beyond the 3 levels of modeling outlined above:

- 1/ Particle Methods
- 2/ Distribution Methods
- 3/ Moment Methods

There exist numerous "hybrid" models that combine features of several methods. Also, moments are typically calculated in particle and distribution methods as diagnostics.

• Hybrids: AF, gyro-kinetic, ....

In general:

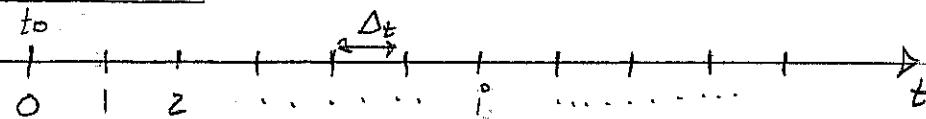
PIC with shaped particles; etc.

- Particle and Distribution methods are appropriate for higher levels of detail
- Moment methods are used for rapid iteration of machine design.
- Even within a single method (particle say) there exist many levels of description.
  - Electromagnetic, electrostatic + many field sol. numerical methods
  - 1d, 2d, 3d, ....
  - !
- Employing a hierarchy of models with full diagnostics allows cross-checking (both in numerics and physics) and aids understanding.
  - No single method is best in all cases.

### §3 Numerical Methods!

General approach is to discretize independent variables in each of the methods and solve for dependant variables which may in cases also be discretized.

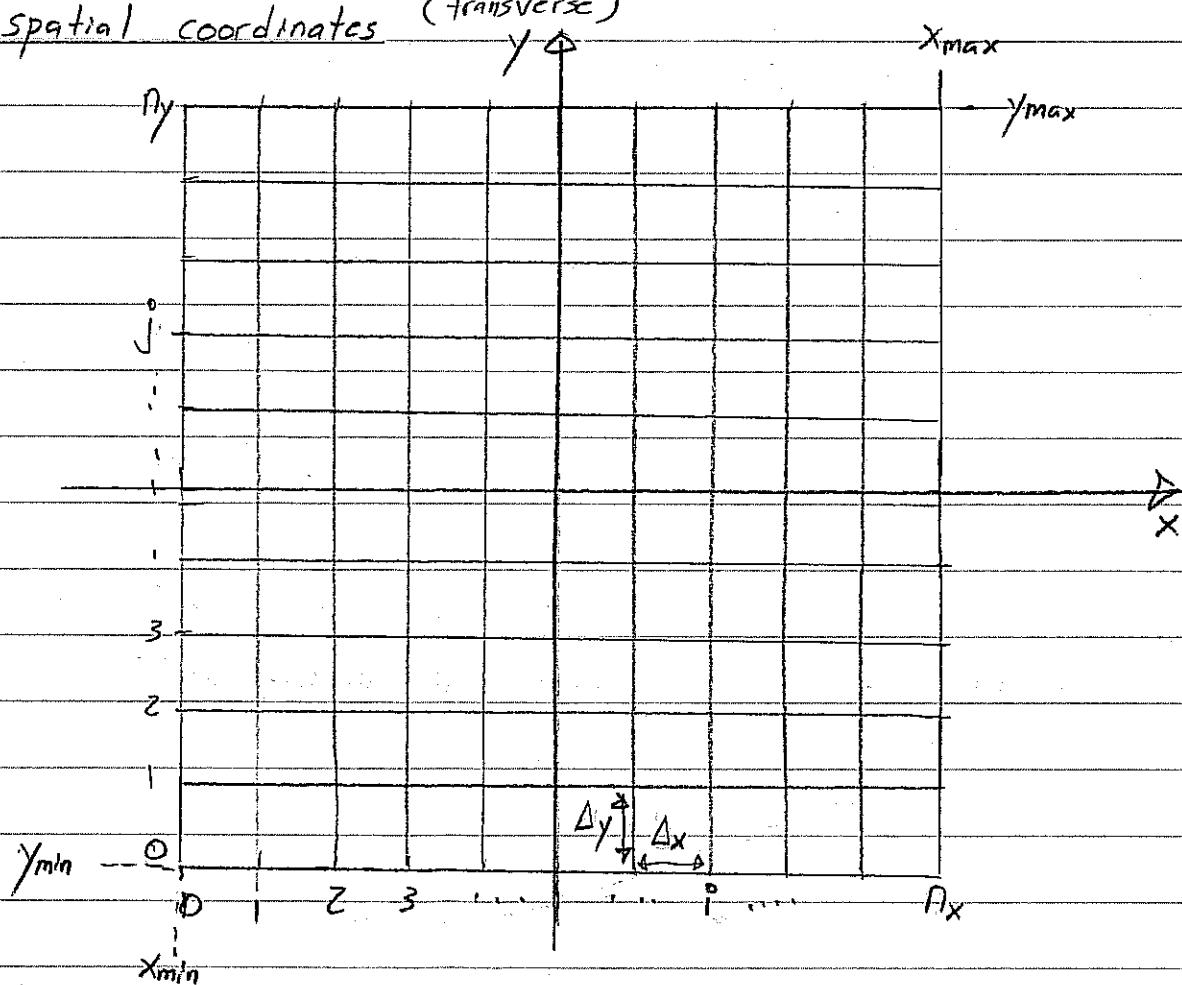
time (or s)



↑  
initial condition

$$t = t_0 + i \Delta t \quad ; \quad i = 0, 1, 2, 3, \dots$$

spatial coordinates (transverse)



$$x_i = x_{\min} + \Delta x \cdot i \quad ; \quad \Delta x = (x_{\max} - x_{\min}) / N_x \quad ; \quad i = 0, 1, 2, \dots, N_x$$

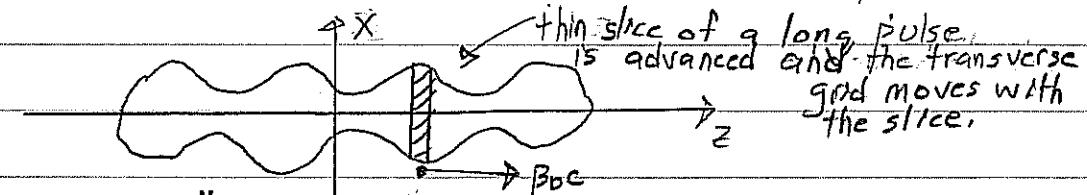
$$y_j = y_{\min} + \Delta y \cdot j \quad ; \quad \Delta y = (y_{\max} - y_{\min}) / N_y \quad ; \quad j = 0, 1, 2, \dots, N_y$$

analogous for 3d, momentum coordinates, etc...

Spatial coordinates (longitudinal)

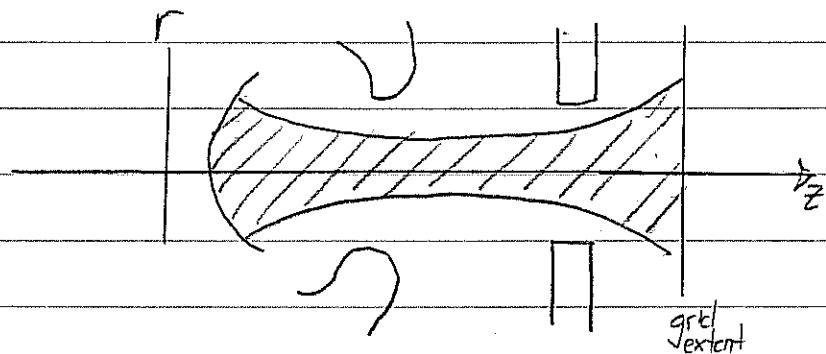
In typical applications may:

- Move a thin "slice" of beam along the axial coordinate  $s$  of a reference particle.



- This "steady flow" approximation is not always possible - some problems are Intrinsic 3D.

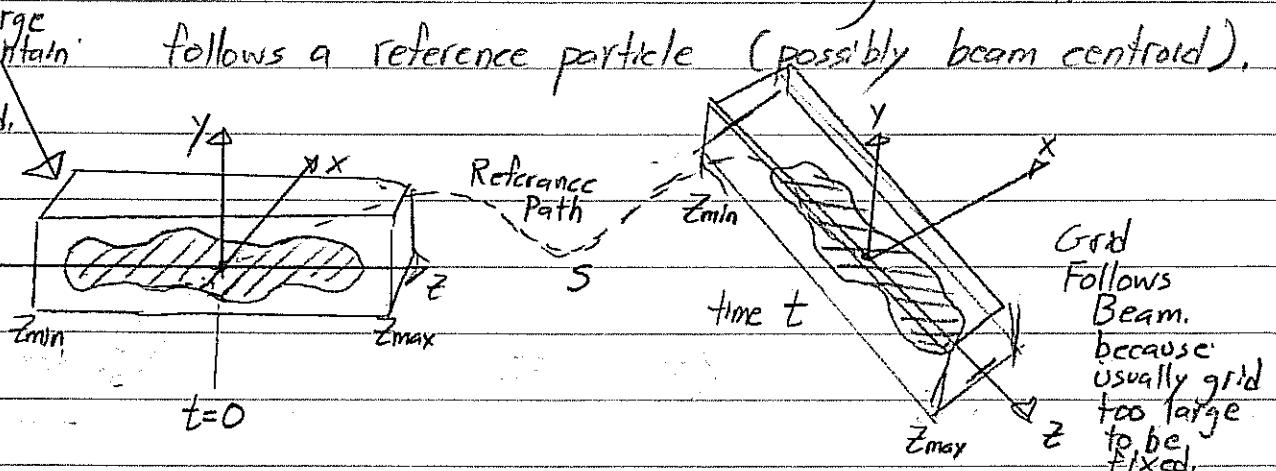
- Simulate a steady state situation like in a mid-pulse diode where the beam fills the grid.



In these situations the mesh is stationary.

- Simulate a 3D beam with a moving mesh that follows a reference particle (possibly beam centroid).

Grid Region large enough to contain 3D beam. Can be resized.



- Grid may be moved in discretized jumps so that applied fields maintain alignment with the grid.

Particle and distribution methods require further levels of discretization in field and other quantities to be rendered practical. Before discussing this we will first discuss basic discrete numerical operations employed. Then we will apply these operations to the simplest case, moment methods, before discussing the more complicated particle and distribution methods.

### § 3 A. Discrete Numerical Operations

Let  $x$  represent a space or time coordinate and  $f(x)$  some continuous function of  $x$ .

$$x_i = x_{\min} + \Delta x \cdot i \quad ; \quad \Delta x = (x_{\max} - x_{\min}) / n_x \quad ; \quad i = 0, 1, 2, \dots, n_x$$

Denote

$$f_i = f(x=x_i) \quad , \text{ etc.}$$

and Taylor expand:

$$f_{i+1} = f_i + \frac{\partial f}{\partial x} \Big|_{x_i} \Delta x + \frac{1}{2!} \frac{\partial^2 f}{\partial x^2} \Big|_{x_i} \Delta x^2 + \frac{1}{3!} \frac{\partial^3 f}{\partial x^3} \Big|_{x_i} \Delta x^3 + \dots$$

$$f_{i-1} = f_i - \frac{\partial f}{\partial x} \Big|_{x_i} \Delta x + \frac{1}{2!} \frac{\partial^2 f}{\partial x^2} \Big|_{x_i} \Delta x^2 - \frac{1}{3!} \frac{\partial^3 f}{\partial x^3} \Big|_{x_i} \Delta x^3 + \dots$$

Derivatives

Low order derivatives follow immediately

$$\text{Forward: } \frac{df}{dx} \Big|_i = \frac{f_{i+1} - f_i}{\Delta x} + O(\Delta x)$$

$$\text{Backward: } \frac{df}{dx} \Big|_i = \frac{f_i - f_{i-1}}{\Delta x} + O(\Delta x)$$

A more accurate, centered, scheme is obtained by subtracting the two expansions! But needs more grid points for accuracy.

$$\text{Centered: } \frac{df}{dx} \Big|_i = \frac{f_{i+1} - f_{i-1}}{2\Delta x} + O(\Delta x^2)$$

The expansions can be relabeled ( $i \rightarrow i+1$ , etc) and the resulting equations analyzed to obtain 5-point and other higher-order forms such as:

$$5 \text{ Point: } \frac{df}{dx} \Big|_i = \frac{f_{i+2} - 8f_{i+1} + 8f_{i-1} - f_{i-2}}{12\Delta x} + O(\Delta x^4)$$

Still higher order forms are possible but rapidly become more cumbersome and require more points. Higher order derivatives can be similarly obtained. For example, adding the two expansions shows:

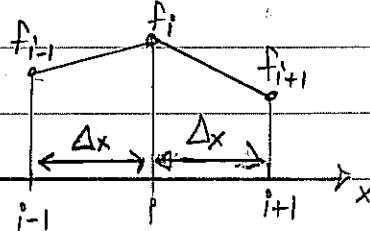
$$\frac{d^2f}{dx^2} \Big|_i = \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} + O(\Delta x^2)$$

## Integrals (Quadrature)

Take  $n_x$  even, then  $\int_{x_{min}}^{x_{max}} dx f(x)$  can be composed as sub-integrals of form:

$$\int_{x_{i-1}}^{x_{i+1}} d\tilde{x} f(\tilde{x})$$

Using a linear approximation: (Trapezoidal Rule)



$$\int_{x_{i-1}}^{x_{i+1}} dx f(x) = \frac{f_{i-1} + 2f_i + f_{i+1}}{2} \Delta x + \mathcal{O}(\Delta x^3)$$

Better approximations can be found (Simpson's Rule) using Taylor series expansions and the previous discrete derivatives:

$$f(x) = f_i + \frac{f_{i+1} - f_{i-1}}{2 \Delta x} x + \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} x^2 + \dots$$

giving:

$$\int_{x_{i-1}}^{x_{i+1}} dx f(x) = \frac{(f_{i+1} + 4f_i + f_{i-1})}{3} \Delta x + \mathcal{O}(\Delta x^5)$$

Numerous other higher order formulas can be derived but are more complicated.

In the examples given, uniform meshes have been employed and formulas are readily generalized to multi-dimensions. In practice, nonuniform meshes are also often used since many applications need resolution concentrated in particular regions. Unfortunately, this makes the differencing formulas more complicated. Particularly with respect to ordering errors. An example of derivative calculations on a nonuniform 1d grid is included in the homework to illustrate methods.

### §3B Numerical Solution of Moment Methods

We now have the tools to discuss the numerical solution of moment methods. The moment equations may always be written as an  $N$ -dimensional set of coupled 1st order ordinary differential equations:

$$\boxed{\begin{aligned}\vec{M} &= (x_c, \langle \hat{x}^2 \rangle, \dots) \\ \frac{d\vec{M}}{ds} &= \vec{F}(\vec{M}, s)\end{aligned}}$$

// Example:

Axissymmetric envelope equation for a continuously focused beam

$$\frac{d^2 R}{ds^2} + k_{po}^2 R - \frac{Q}{R} - \frac{Ex^2}{R^3} = 0 \quad , \quad k_{po}, Q, Ex^2 \text{ constants.}$$

$$\frac{d}{ds} \begin{bmatrix} R' \\ R \end{bmatrix} = \begin{bmatrix} -k_{po}^2 R + \frac{Q}{R} + \frac{Ex^2}{R^3} \\ R' \end{bmatrix}$$

///

Euler's Method:

Apply the forward diff. formula

$$\frac{d\vec{M}_i}{ds} = \frac{\vec{M}_{i+1} - \vec{M}_i}{\Delta s} + \mathcal{O}(\Delta s) = \vec{F}(\vec{M}_i, s_i)$$

1st  
order  
Euler  
Method:

$$\boxed{\vec{M}_{i+1} = \vec{M}_i + \Delta s \vec{F}(\vec{M}_i, s_i) + \mathcal{O}(\Delta s^2)}$$

The moments can then be advanced from their initial values.

Note that for  $N_s$  steps will lead to a total error  $\sim N_s \cdot \mathcal{O}(\Delta s^2) \sim (\delta_{\max} - \delta_{\min}) \cdot \mathcal{O}(\Delta s^2) \sim \mathcal{O}(\Delta s)$ .

- Error decreases only linearly with step size.

- = Numerical work for each step is only 1 evaluation of  $\vec{F}$ .

### Definition

A discrete advance with error  $\mathcal{O}(\Delta s^n)$  is an  $n-1$ 'th order method.

Thus Euler's method is a 1st order method.

Higher order methods are generally used for ODE's

Runge-Kutta Method:

$$\frac{d\vec{M}}{ds} = \vec{F}(\vec{M}, s)$$

Integrate:

$$\therefore \vec{M}_{i+1} = \vec{M}_i + \int_{s_i}^{s_{i+1}} ds \vec{F}(\vec{M}, s)$$

Approximate  $\vec{F}$  by a Taylor expansion through the midpoint of the interval. The linear term integrates to zero leaving:

$$\vec{F}(\vec{M}, s) = \vec{F}(\vec{M}_{i+\frac{1}{2}}, s_{i+\frac{1}{2}}) + \frac{d\vec{F}}{ds}\Big|_{i+\frac{1}{2}} \cdot s + \dots$$

$$\Rightarrow \vec{M}_{i+1} = \vec{M}_i + \vec{F}(\vec{M}_{i+\frac{1}{2}}, s_{i+\frac{1}{2}}) \cdot \Delta s + \mathcal{O}(\Delta s^3)$$

Note: only need  $\vec{M}_{i+\frac{1}{2}}$  to  $\mathcal{O}(\Delta s^2)$  for  $\mathcal{O}(\Delta s^3)$  accuracy. Apply Euler's method for the 2-step procedure:

$$\vec{k} \equiv \Delta s \vec{F}(\vec{M}_i, s_i)$$

2nd Order

Runge-Kutta

Method:  $\vec{M}_{i+1} = \vec{M}_i + \Delta s \cdot \vec{F}(\vec{M}_i + \frac{\vec{k}}{2}, s_i + \frac{\Delta s}{2}) + \mathcal{O}(\Delta s^3)$

- Requires 2 evaluations of  $\vec{F}$  per advance.
- 2nd order accurate in  $\Delta s$ .

Higher order Runge-Kutta Schemes are analogously derived from various quadrature formulas. Many such formulas are derived in standard numerical texts.

- Typically, methods with local error  $\mathcal{O}(\Delta t^{N+1})$  will require  $N$  evaluations of  $\vec{F}$ .

For beam physics, many methods are employed to advance moments and particle orbits. A general survey of these methods is beyond the scope of this lecture. But some general comments can be made:

- Many high-order methods with adaptive stepsize exist that refine accuracy to specified tolerances and are optimized for specific classes of equations.
- Choice of methods often involves numerical work and stability considerations.
- Certain methods can be formulated to exactly preserve relevant single-particle invariants.
  - "Symplectic" methods preserve Hamiltonian structure of dynamics.
- Accelerator problems can be demanding due to multiple frequency scales and long tracking times/distances.

Numerical stability is key - but is difficult to analyze in general.

One simple test often used is a "Numerical Reversibility" check. In this method, the final value of an advance is used as an initial condition. Then the problem is run backwards to the original start point and deviations analyzed.

- Often a stringent test of accuracy.
- Will ultimately fail due to roundoff errors and cases where there is a sensitive dependence on initial conditions.
- Orbits can be wrong but qualitatively right .... we will quantify this notion better later. So lack of full convergence does not necessarily mean that useless results will be obtained.

We now briefly overview an application of Moment equations, namely the KV envelope equations, to a practical high current transport lattice that was designed for heavy ion fusion applications at Lawrence Berkeley National Laboratory.

# Moment Equation Application: Pep. RMS Envelope Equations

Neglect image charges and nonlinear self-fields (emittance constant) to obtain moment equations for the evolution of the beam envelope radii

$$\begin{aligned} \frac{d^2 r_x}{ds^2} + \kappa_q r_x - \frac{2Q}{r_x + r_y} - \frac{\epsilon_x^2}{r_x^3} &= 0 \\ \frac{d^2 r_y}{ds^2} - \kappa_q r_y - \frac{2Q}{r_x + r_y} - \frac{\epsilon_y^2}{r_y^3} &= 0 \end{aligned}$$

statistical beam envelope radii

$$r_x = 2\sqrt{\langle x^2 \rangle}$$

$$r_y = 2\sqrt{\langle y^2 \rangle}$$

$$Q = \frac{qI}{2\pi\epsilon_0 mc^3 \gamma_b^3 \beta_b^3}$$

... Dimensionless Perveance  
measures space-charge strength

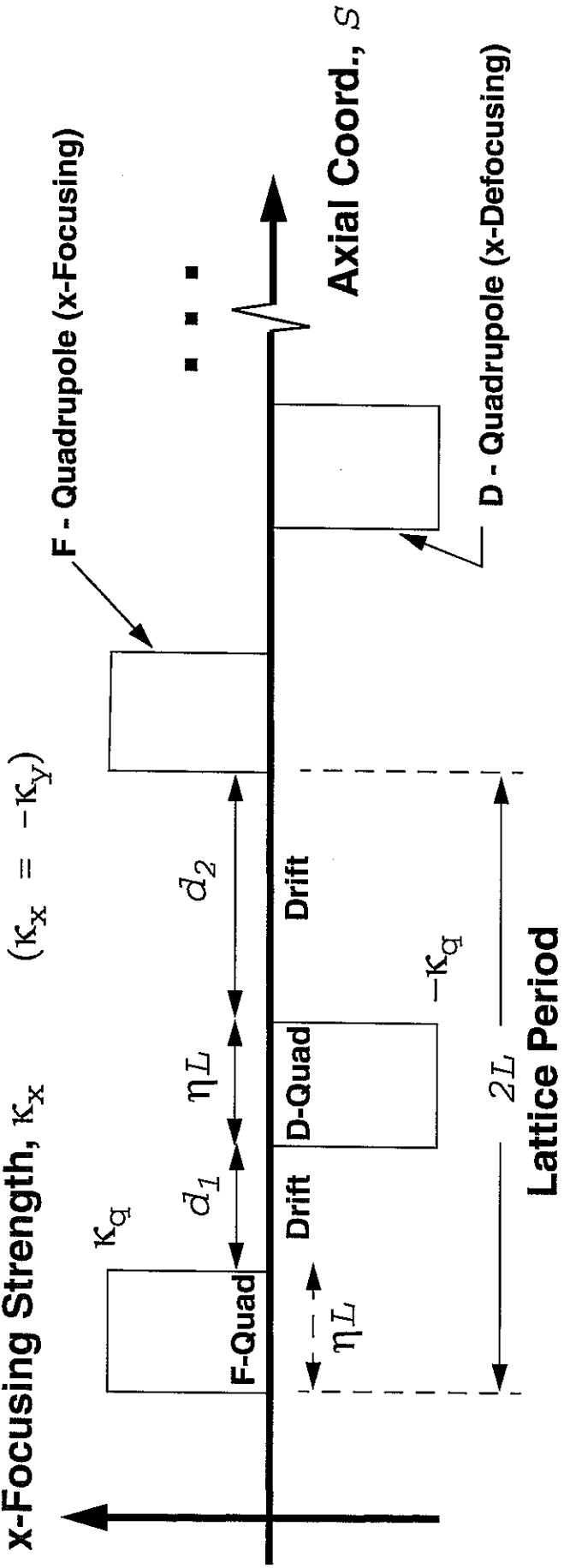
$$\epsilon_x = 4[\langle x^2 \rangle \langle x'^2 \rangle - \langle xx' \rangle^2]^{1/2}$$

... RMS Edge Emittance  
measures x-x' phase-space area  
~(beam size)sqrt(thermal temperature)

(  $\epsilon_{xn} = \gamma_b \beta_b \epsilon_x$  normalized )

The matched beam solution together with parametric constraints from engineering, higher-order theory, and simulations are used to design the lattice

# Alternating Gradient Focusing Lattice with Syncopation



**Drifts:**

$$d_1 = \alpha(1-\eta)2L$$

$$d_2 = (1-\alpha)(1-\eta)2L$$

**Focusing Strength:**

$$\kappa_q = \begin{cases} \frac{1}{[B\rho]} \left| \frac{dB_x}{dy} \right| & , \text{ Magnetic Quadrupole} \\ \frac{1}{[B\rho]V_b} \left| \frac{dE_x}{dx} \right| & , \text{ Electric Quadrupole} \end{cases}$$

S. M. Land

12/

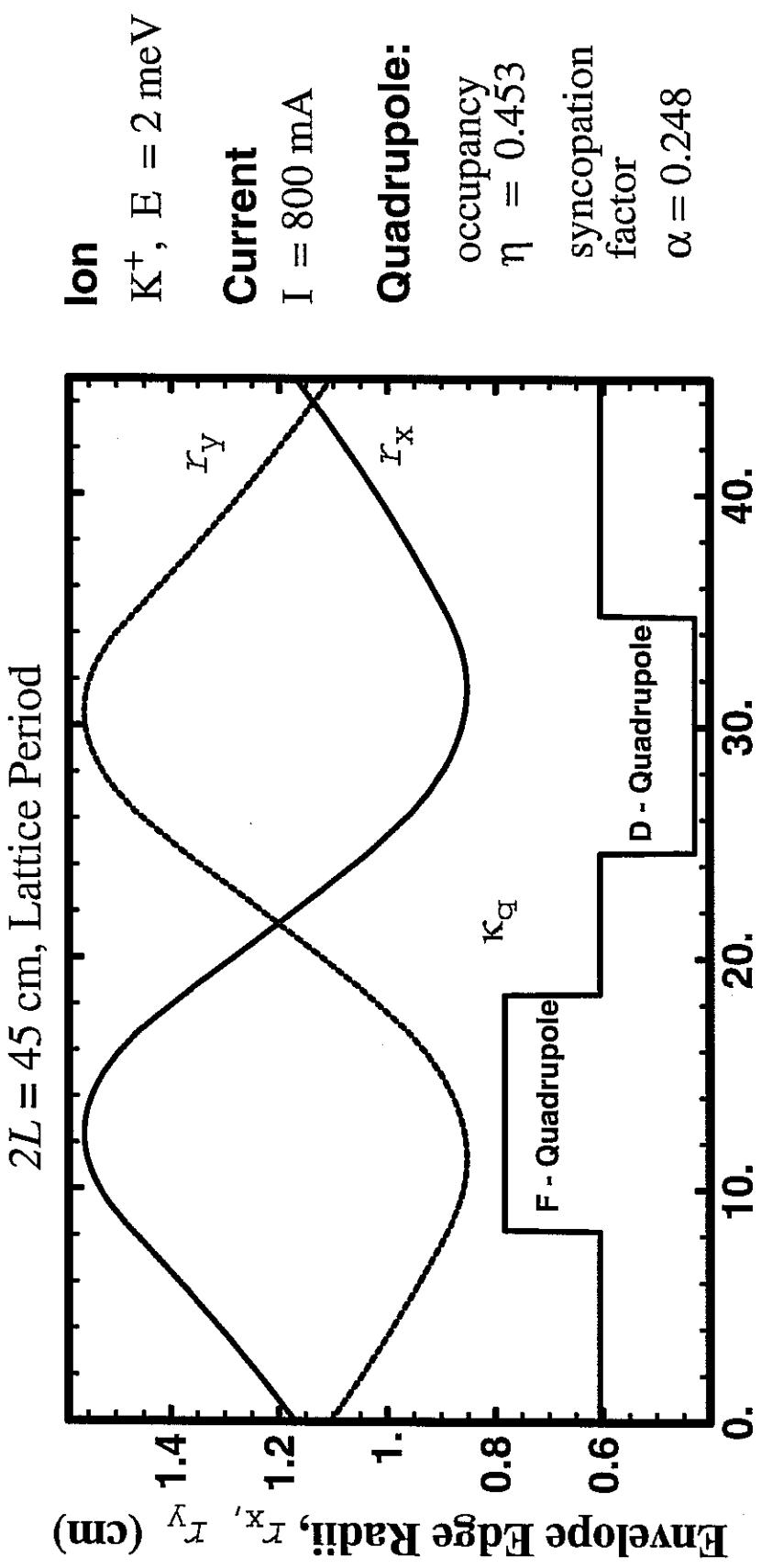
- $\eta = \text{Quadrupole Occupancy}$  ( $0 < \eta < 1$ )
- $\alpha = \text{Syncopation Factor}$  ( $0 < \alpha < 1$ )
- $\alpha = 1/2 \Rightarrow \text{Symmetric FODO}$ ,  $d_1 = d_2$
- $\alpha = 0, 1 \Rightarrow F \text{ touches D}$

**Rigidity**

$$[B\rho] = m\gamma_b V_b / q$$

**Electric Quadrupole**

# Matched Envelope Solution and Particle Phase Advances



**Low Emittance Case:**  $\varepsilon_{x,rms} = 33.0$  mm-mrad, rms edge

$$\overline{r_x} = \overline{r_y} = 1.184 \text{ cm}$$

$$\begin{aligned} \text{Max}[r_x] &= \text{Max}[r_y] = 1.558 \text{ cm} & \text{Min}[r_x'] &= -\text{Min}[r_y'] = 38.9 \text{ mrad} & \sigma_{\beta 0} &= 80^\circ / \text{lattice-period} \\ \text{Min}[r_x] &= \text{Min}[r_y] = 0.853 \text{ cm} & \text{Max}[r_y'] &= -\text{Min}[r_x'] = 58.0 \text{ mrad} & \sigma_\beta &= 6.88^\circ / \text{lattice-period} \end{aligned}$$

**High Emittance Case:**  $\varepsilon_{x,rms} = 100.0$  mm-mrad, rms edge

$$\overline{r_x} = \overline{r_y} = 1.218 \text{ cm}$$

$$\begin{aligned} \text{Max}[r_x] &= \text{Max}[r_y] = 1.605 \text{ cm} & \text{Min}[r_x'] &= -\text{Min}[r_y'] = 40.2 \text{ mrad} & \sigma_{\beta 0} &= 80^\circ / \text{lattice-period} \\ \text{Min}[r_x] &= \text{Min}[r_y] = 0.875 \text{ cm} & \text{Max}[r_y'] &= -\text{Min}[r_x'] = 60.0 \text{ mrad} & \sigma_\beta &= 19.8^\circ / \text{lattice-period} \end{aligned}$$

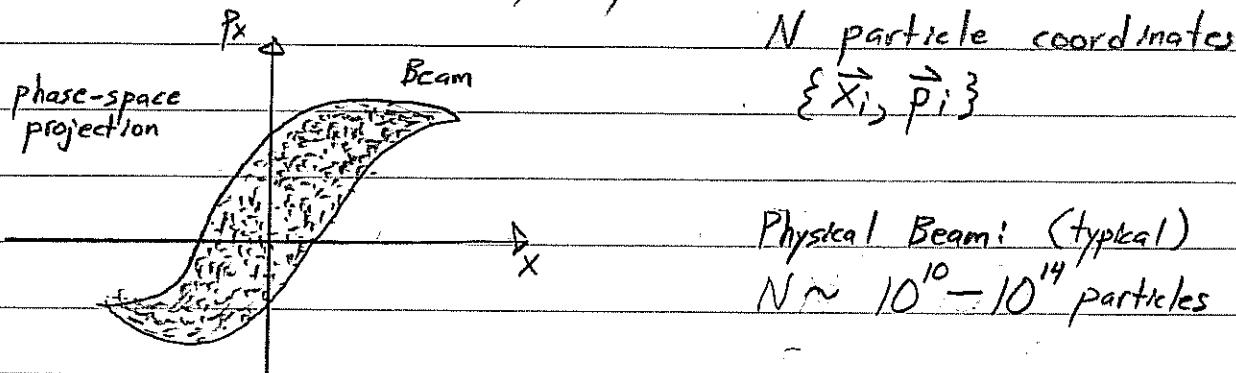
§4

## Numerical Solution of Particle and Distribution Methods

Particle methods - generally not used at high space-charge intensity

Distribution methods - preferred - mainly PIC - for high space-charge. We will motivate why now.

Particle Methods - Why they are not a good choice:



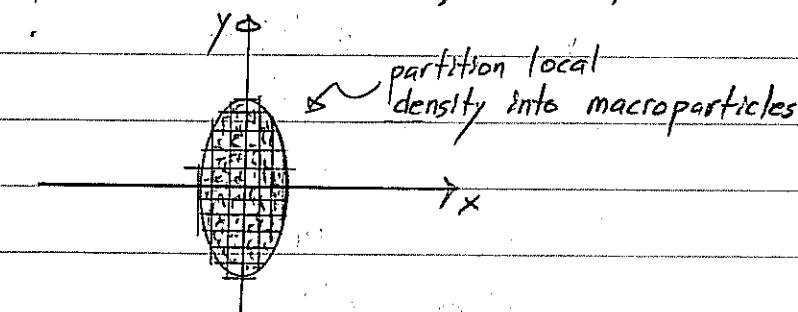
Physical Beam: (typical)

$N \sim 10^{10} - 10^{14}$  particles

Although larger problems are possible every year with more powerful computers, processor speed and memory limitations presently limit us to

$$N \lesssim 10^8 \text{ particles.}$$

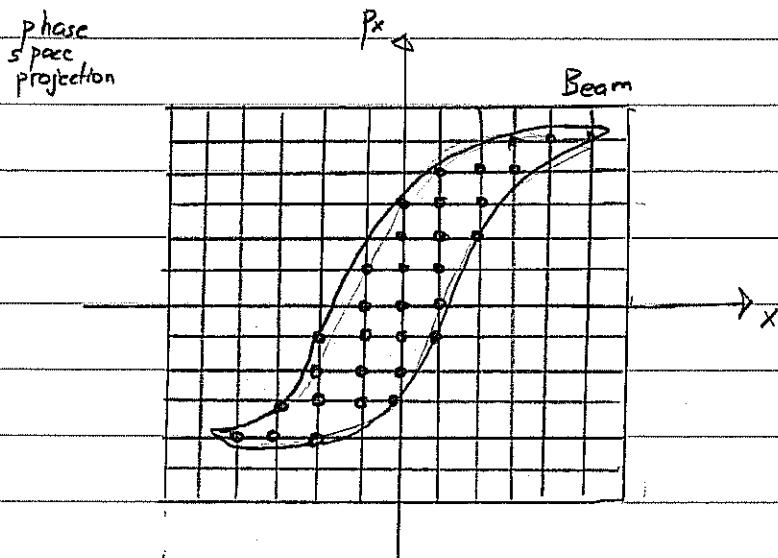
Represent the beam by "macroparticles" and advance macroparticles in time.



- Same q/m ratio as real particle  $\Rightarrow$  same single particle dynamics in applied field.
- Results in more collisions due to macroparticles having more close approaches.
  - Enhanced collisionality is unphysical
  - Controlled by smoothing the macroparticle interaction with the self-field. More on this later!

Distribution Methods

Vlasov as an example.

Discrete grid points  $\{\vec{x}_i, \vec{p}_i\}$ Advanced distribution  $f(\vec{x}, \vec{p}, t)$  at discrete grid points in time.

- Continuum distribution advanced on a discrete phase-space mesh.
  - Extreme memory for high resolution. Ex.  
for 4d  $x-P_x, y-P_y$  with 100 mesh  
points on each axis  $\Rightarrow 100^4 = 10^8$   
values to store in fast memory.
- Discretization errors can lead to aliasing and unphysical behaviour (negative probability etc.).

Both particle and distribution methods can be broken up into 2 basic parts:

- 1) Moving particles or distribution evaluated at grid points through a finite time (or axial space) step.
- 2) Calculation of beam self-fields consistently with the distribution of particles.

In both methods, significant fractions of the run time may also be devoted to diagnostics:

- Moments - can be computationally inexpensive and may be accumulated frequently for evolution of evolution "histories".
- Phase space projections ("snapshot" in time)
- Fields (snapshot in time)

Diagnosers are also critical!

- Without appropriate diagnostics runs are useless even if correct.
- Most accumulate and present mass data in an understandable format.

Significant source code development may also be devoted to creating (loading) the initial distribution of particles to simulate.

- Such parts of the simulation will usually only take a small fraction of total run time.

## Particle Methods - Integration of the Equations of Motion

Higher order methods require more storage and numerical work per time step

- Fieldsolves are expensive, especially in 3D, and several fieldsolves per step can be necessary for higher order accuracy.

for self-consistent space-charge.

Typically, low-order methods are used. The "leapfrog" method is the most common.

- Only need to store prior position and velocity
- One fieldsolve per timestep.

Illustrate for

Nonrelativistic equations of motion:

$$m \frac{d\vec{v}}{dt} = \vec{F} = g(\vec{E} + \vec{v} \times \vec{B})$$

$$\frac{d\vec{x}}{dt} = \vec{v}$$

Leapfrog advance (time centered)

$$1) m \cdot \frac{\vec{v}_{i+1/2} - \vec{v}_{i-1/2}}{\Delta t} = \vec{F}_i$$

Note:  $\vec{x}$  and  $\vec{v}$

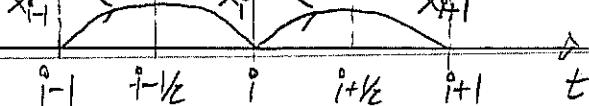
$$2) \frac{\vec{x}_{i+1} - \vec{x}_i}{\Delta t} = \vec{v}_{i+1/2}$$

advanced out of phase.

Velocity:



Position:



For purposes of analysis of the properties of the leap-frog method it is often convenient to write the map in an alternative form:

$$\frac{\vec{x}_{i+1} - \vec{x}_i}{\Delta t} = \vec{v}_{i+1/2}$$

$$i \rightarrow i-1 \quad \frac{\vec{x}_i - \vec{x}_{i-1}}{\Delta t} = \vec{v}_{i-1/2}$$

Subtract the equations and apply the other formula:

$$\Rightarrow m \frac{\vec{v}_{i+1/2} - \vec{v}_{i-1/2}}{\Delta t} = m \frac{\vec{x}_{i+1} - 2\vec{x}_i + \vec{x}_{i-1}}{\Delta t^2} = \vec{F}_i$$

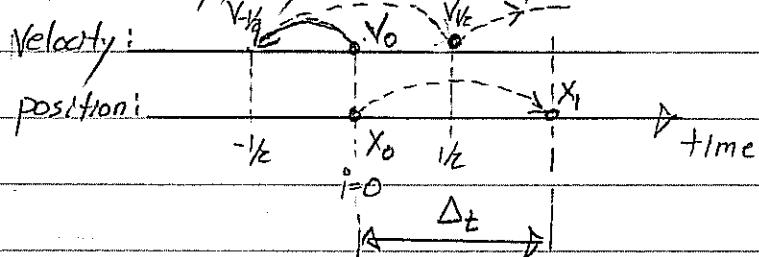
Note:  $\frac{\partial^2 f}{\partial x^2}|_{x_i} = \frac{f_{i+1} - 2f_i + f_{i-1}}{\Delta x^2} + O(\Delta x)$   
 $\Rightarrow \vec{x}_{i+1}$  fixed from  
 $\vec{x}_i, \vec{x}_{i-1}, \vec{F}_i$  to  $O(\Delta t^4)$

Synchronization and desynchronization:

Since  $\vec{x}$  and  $\vec{v}$  are not evaluated at the same time

In the leapfrog method:

- Initial conditions must be adjusted to start the advance. Typically,  $\vec{v}$  is pushed back a half cycle.



- When evaluating diagnostic quantities such as moments the particle coordinates and velocities should first be synchronized using a similar procedure.

Another complication is in evaluation of the force

which has velocity dependence when magnetic fields

are present since  $\vec{x}$  and  $\vec{v}$  are advanced out of phase  
in the leap-frog method.

$$\vec{F} = q \vec{E} + q \vec{v} \times \vec{B}$$

Note!

Velocity term

- Electric field  $\vec{E}$  accelerates.

- Magnetic field  $\vec{B}$  bends particle trajectory without change in speed  $\vec{v}$ .

A commonly implemented time centered scheme for this is the following 3-step "Boris" method:

1) Half-step acceleration in electric field

$$\vec{V}_{i+1/2}^{(1)} = \vec{V}_{i-1/2} + \frac{q}{m} \vec{E}_i \cdot \frac{\Delta t}{2}$$

2) Rotation in magnetic field. Here choose coordinates such that  $\vec{B}_i = B_i \hat{z}$ ,  $\omega_{ci} = q B_i / m$

$$\text{II } B_i \parallel V_{z,i+1/2}^{(2)} = V_{z,i+1/2}^{(1)} \quad \begin{array}{l} \text{Parallel Magnetic Field to } \vec{v} \\ \text{does not bend trajectory} \end{array}$$

$$\perp B_i \quad \begin{pmatrix} V_{x,i+1/2}^{(2)} \\ V_{y,i+1/2}^{(2)} \end{pmatrix} = \begin{pmatrix} \cos(\omega_{ci} \Delta t) & \sin(\omega_{ci} \Delta t) \\ -\sin(\omega_{ci} \Delta t) & \cos(\omega_{ci} \Delta t) \end{pmatrix} \begin{pmatrix} V_{x,i+1/2}^{(1)} \\ V_{y,i+1/2}^{(1)} \end{pmatrix}$$

3) Half-step acceleration in electric field

$$\vec{V}_{i+1/2} = \vec{V}_{i+1/2}^{(3)} = \vec{V}_{i+1/2}^{(2)} + \frac{q}{m} \vec{E}_i \cdot \frac{\Delta t}{2}$$

Complication! On startup how does one generate the out of phase  $\vec{x}, \vec{v}$  advance from initial conditions?

- Calculate  $\vec{E}, \vec{B}$  with initial conditions.
- Move  $\vec{v}$  backward a half-step ( $-\Delta t/2$ )
  - Rotate with  $\vec{B}$  a half-step.
  - Decelerate a half-step in  $\vec{E}$ .

Similar comments hold for synchronization of  $\vec{x}, \vec{v}$  for diagnostic accumulation.

Now we will look at the numerical properties of the leapfrog advance cycle.

- Only use a simple example to illustrate issues.

# Leapfrog Advance Errors and Numerical Stability

To better understand the leapfrog method consider the simple harmonic oscillator:

$$\frac{d^2x}{dt^2} = -\omega^2 x \quad \omega = \text{const.}$$

Exact

$$\text{Solution} \Rightarrow x = C_0 \cos \omega t + C_1 \sin \omega t \quad C_0, C_1 \text{ constants}$$

$$= x_0 \cos(\omega t + \psi_0) \quad \text{set from initial cond.}$$

In the leapfrog method:

$$\frac{x_{p+1} - 2x_p + x_{p-1}}{\Delta t^2} = -\omega^2 x_p$$

Try a solution of form

$$x_p = C_0 \cos \tilde{\omega} t_p + C_1 \sin \tilde{\omega} t_p$$

Substitute

$$x_p = C e^{j \tilde{\omega} t_p \Delta t} \quad j = \sqrt{-1} \quad \text{use } j \text{ to not confuse with other index choices.}$$

$$\Rightarrow e^{j \tilde{\omega} \Delta t} - 2 + e^{-j \tilde{\omega} \Delta t} = -\omega^2 \Delta t^2$$

$$2 - 2 \cos(\tilde{\omega} \Delta t) = \omega^2 \Delta t^2$$

$$\sin^2\left(\frac{\tilde{\omega} \Delta t}{2}\right) = \frac{\omega^2 \Delta t^2}{4} \quad \text{take + root}$$

$$\Rightarrow \sin\left(\frac{\tilde{\omega} \Delta t}{2}\right) = \frac{\omega \Delta t}{2}$$

This has solutions for  $\omega \Delta t < 2$  and it is straightforward to show (expansion) that for  $\omega \Delta t$  small:

$$\tilde{\omega} \Delta t = \omega \Delta t + \frac{(\omega \Delta t)^3}{24} + \mathcal{O}(\omega \Delta t^5)$$

It follows immediately for the leapfrog method:

- For  $\omega\Delta t \ll 1$  the method is stable
- There is no amplitude error in the integration
- For  $\omega\Delta t \ll 1$ , the phase error is
  - Actual phase:  $\Psi = \omega i \Delta t$
  - Simulated phase:  $\tilde{\Psi} = \tilde{\omega} i \Delta t \approx \omega i \Delta t + (\omega \Delta t)^3 \frac{24}{i}$
  - Error phase:  $\Delta \Psi = \tilde{\Psi} - \Psi = (\omega \Delta t)^3 \frac{24}{i}$

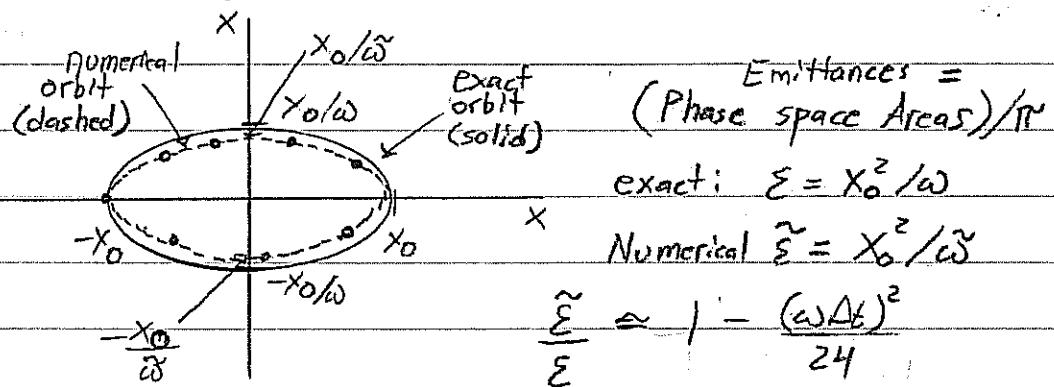
Note:  $i$  to get to a fixed time  $\sim 1/\Delta t$   
and therefore phase errors decrease as  $O(\Delta t^2)$

// Example:  $\omega = 2\pi/T$

$\Delta t = 0.1 T$ : steps for a  $\pi$  phase error  $\approx 24\pi/(0.1 \cdot 2\pi)^3 \approx 304 \Rightarrow \sim 30.4$  cycles

$\Delta t = 0.01 T$ : steps for a  $\pi$  phase error  $\approx 24\pi/(0.01 \cdot 2\pi)^3 \approx 3.04 \times 10^5 \Rightarrow \sim 3,040$  cycles

### Contrast: Numerical and Actual Orbit



The numerical orbit conserves phase space area regardless of the number of steps taken! The slight differences between the numerical and actual orbit can be removed by rescaling  $\omega$  to account for the discrete step.

- More general analysis of the leapfrog map shows it has "symplectic" structure - meaning it preserves the Hamiltonian nature of the dynamics.
- Symplectic methods are important for long tracking problems (typical in accelerators) to obtain the right orbit structure - Runge-Kutta is not symplectic

## Particle Methods - Field solution

The self-consistent calculation of self-fields is of primary importance to accurately simulate intense particle beams. Techniques outlined here are also applicable to distribution methods.

$$\vec{F} = q\vec{E} + q\vec{v} \times \vec{B}$$

Fields can be resolved into externally applied and self (i.e., beam generated) components:

$\vec{E} = \vec{E}_a + \vec{E}_s$	$\vec{E}_a, \vec{B}_a$ : Applied
$\vec{B} = \vec{B}_a + \vec{B}_s$	$\vec{E}_s, \vec{B}_s$ : Self

- $\vec{E}_a, \vec{B}_a$  applied fields generated from magnets and electrodes
  - sometimes calculated at high resolution in external codes and imported or specified via analytic formulae
  - sometimes calculated from code field solve via applied charges and currents and boundary conditions
- $\vec{E}_s, \vec{B}_s$  self fields generated from beam charges and currents.
  - At high beam intensities can be a large fraction (on average) of the applied fields
  - Important to calculate with realistic boundary conditions.

For simplicity, we restrict analysis to electrostatic problems to illustrate methods:

$$\vec{B} = \vec{B}_q \quad (\text{specified via another code or theory})$$

$$\vec{E} = \vec{E}_q + \vec{E}_s \quad (\vec{E}_q \text{ due to biased electrodes and } \vec{E}_s \text{ due to beam space-charge})$$

The Maxwell equations to be solved for  $\vec{E}$  are

$\nabla \cdot \vec{E} = \frac{f}{\epsilon_0}$	+ Boundary conditions on $\vec{E}$ .
$\nabla \times \vec{E} = 0$	

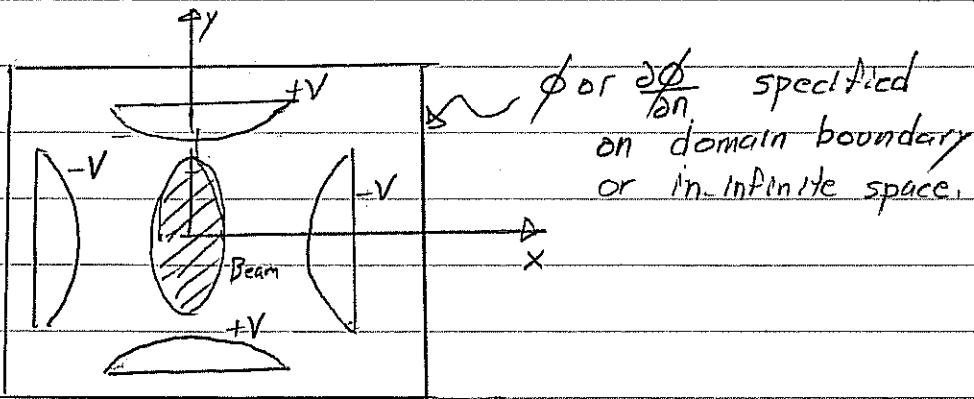
$\nabla \times \vec{E}$  implies that we can always take

$$\vec{E} = -\nabla \phi$$

and we have

$\nabla^2 \phi = -\frac{f}{\epsilon_0}$	+ Boundary conditions on $\phi$
$\vec{E} = -\nabla \phi$	

As an explicit example, it might be necessary to solve (2D) problems like:



Formally, the solution to  $\phi$  can be constructed with a Green's Function: (Illustrated for Dirichlet Boundary Conditions)

$$\nabla^2 G(\vec{x}, \vec{x}') = -4\pi \delta(\vec{x} - \vec{x}')$$

$$G(\vec{x}, \vec{x}') \Big|_{\vec{x}' \text{ on boundaries}} = 0$$

which gives

$$\vec{E}_S = - \frac{\partial \phi_S}{\partial \vec{x}}$$

$$\vec{E}_q = -\frac{\partial \phi_q}{\partial \vec{x}}$$

- $\phi_a$  can be calculated in advance and need not be recalculated.
    - may be analytical in simple situations.

Let:  $q_m, \vec{x}_i$  = macro-particle charge and coordinate  
 $N_p$  = Macro particle number.

$$\phi_s = \int d\vec{x}' \frac{f(\vec{x}')}{4\pi\epsilon_0} G(\vec{x}, \vec{x}') = \sum_{i=1}^{N_p} \int d\vec{x}' \frac{g_i \delta(\vec{x} - \vec{x}_i)}{4\pi\epsilon_0} G(\vec{x}, \vec{x}')$$

$$\phi_s = \sum_{i=1}^{N_p} \frac{q_m}{4\pi\varepsilon_0} G(\vec{x}, \vec{x}_i)$$

Then the field at the  $j$ th macro-particle is

$$\vec{E}_j = \left. \frac{\partial \phi_s}{\partial \vec{x}} \right|_{\vec{x}=\vec{x}_j} = \frac{g_m}{4\pi\epsilon_0} \sum_{i=1, i \neq j}^{N_p} \frac{\partial G(\vec{x}, \vec{x}_i)}{\partial \vec{x}} \Big|_{\vec{x}=\vec{x}_j}$$

This, in general, will be a numerically intensive expression to evaluate at each macroparticle.

- $N_p(N_p-1)$  terms to evaluate and  $G$  itself will in general be complicated, and may require many costly numerical operations for each term.
- Small number of macroparticles that this procedure may be carried out for will result in a noisy field
  - Enhanced, unphysically high, close approaches (collisions) can change the physics.
- Special "fast multipole" methods based on Greens' functions can reduce the scaling to  $\sim N_p$  or  $\sim N_p \ln N_p$ . However the coefficient is large and smoothing is not easily implemented, usually rendering such methods inferior to gridded methods to be covered shortly.

// Example - Self Fields in free space:

$$G(\vec{x}, \vec{x}') = \frac{1}{|\vec{x} - \vec{x}'|} ; \quad \vec{E}_j = \frac{g_m}{4\pi\epsilon_0} \sum_{i=1, i \neq j}^{N_p} \frac{(\vec{x}_j - \vec{x}_i)}{|\vec{x}_j - \vec{x}_i|^3}$$

An alternative Procedure is Needed to:

- 1) Calculate fields efficiently
- 2) Smooth interactions to compensate for limited particle numbers.

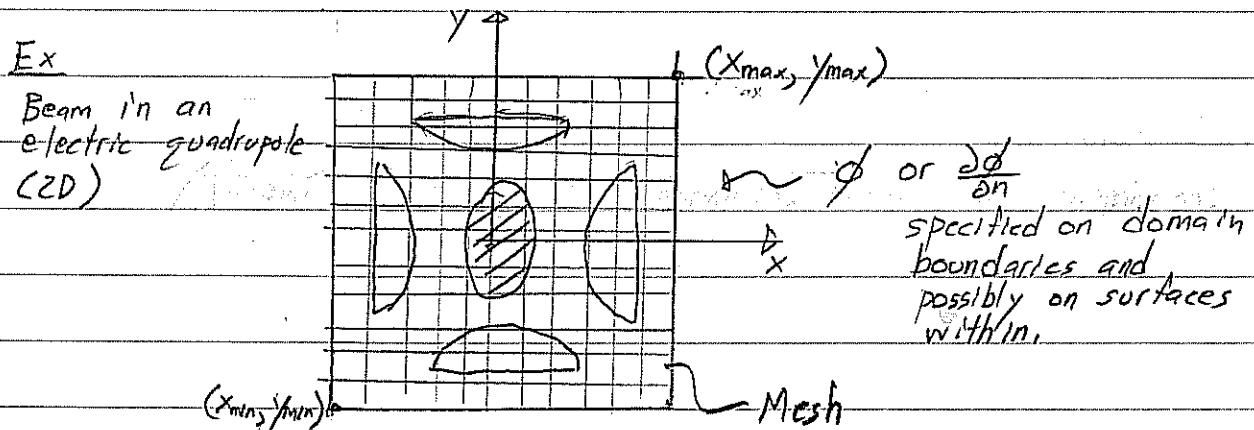
Approach:

Solve the Maxwell Equations on a discrete spatial grid and then smooth the interactions calculated from the gridded field. We will outline this method for 2D interactions, 3D is analogous.

$$x_i = x_{\min} + \Delta x \cdot i \quad \Delta x = (x_{\max} - x_{\min}) / n_x \quad i = 0, 1, 2, \dots, n_x$$

$$y_j = y_{\min} + \Delta y \cdot j \quad \Delta y = (y_{\max} - y_{\min}) / n_y \quad j = 0, 1, 2, \dots, n_y$$

$$\begin{aligned} \vec{E}_{ij} &= \vec{E}(x_i, y_j) \\ \phi_{ij} &= \phi(x_i, y_j) \\ p_{ij} &= P(x_i, y_j) \end{aligned} \quad \left. \right\} \text{Field components, potential, and charge are gridded}$$



Note:  $p_{ij}$  must be calculated from macro-particle not necessarily on grid points. Also fields will ultimately be needed at macro-particle coordinates not on the grid. These issues will be covered later under "particle weighting"

For low order differencing, the Poisson Equation becomes:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = -\frac{p_{i,j}}{\epsilon_0}$$

With the gridded field components calculated as:

$$E_{x,j} = \frac{\phi_{i+1,j} - \phi_{i-1,j}}{2\Delta x}$$

$$E_{y,j} = \frac{\phi_{i,j+1} - \phi_{i,j-1}}{2\Delta y}$$

Boundary conditions must also be incorporated as equations of constraint.

1) Dirichlet Conditions:  $\phi$  specified on surfaces.

Example:  $\phi = V = \text{const}$  at right grid edge

$$\phi_{N_x,j} = V = \text{const}$$

$i = N_x - 1$  cell(s)

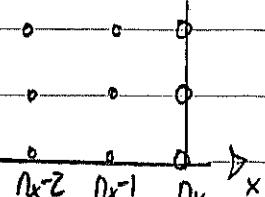
$$\frac{V - 2\phi_{N_x-1,j} + \phi_{N_x,j}}{\Delta x^2} + \frac{\phi_{N_x-1,j+1} - 2\phi_{N_x,j} + \phi_{N_x,j-1}}{\Delta y^2} = -\frac{p_{N_x-1,j}}{\epsilon_0}, \text{ etc.}$$

2) Neumann Conditions:  $\partial\phi/\partial n$  specified on surfaces.

Example:  $\frac{\partial\phi}{\partial x} = G = \text{const}$  at right grid edge

$$\frac{\phi_{N_x-1,j} - \phi_{N_x,j}}{\Delta x} = G$$

$$\frac{\partial\phi}{\partial n} = G$$



The Finite differenced Poisson equation and the boundary conditions can be expressed in Matrix Form:

$$\boxed{\bar{M} \cdot \vec{\phi} = \vec{S}}$$

$\bar{M}$  = coefficients matrix from local finite differences

This matrix will be sparse (i.e., most elements zero)

$\vec{\phi}$  = vector of potentials at grid points.

$\vec{S}$  = "source" terms resulting from beam, charge deposited on the grid ( $p_{ij}$ ) and known potentials from "boundary condition constraints,

Formal solution found by Matrix inversion:

$$\boxed{\vec{\phi} = \bar{M}^{-1} \cdot \vec{S}}$$

Direct Inversion of  $\bar{M}^{-1}$  is not practical due to large dimension of problem. However:

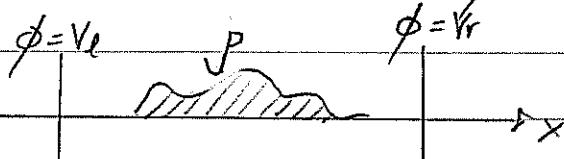
- $\bar{M}$  will in general be very sparse due to the use of local, low-order finite differencing,
- Many fast, numerically efficient inversion methods exist for sparse matrices,
  - Specific method best used depends on type of differencing and boundary conditions.

To illustrate this procedure, consider a simple 1-d example with Dirichlet boundary conditions.

$$\frac{d^2\phi}{dx^2} = -\frac{P}{\epsilon_0}$$

$$\phi(x_e) = V_e \quad \text{left B.C.}$$

$$\phi(x_r) = V_r \quad \text{right B.C.}$$



$$\text{Discretize!} \quad x_i \quad i=0 \quad \dots \quad x_r \quad i=n_x$$

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{\Delta x^2} = -\frac{P_i}{\epsilon_0}$$

$$x_i = x_e + \Delta x \cdot i, \quad \Delta x = (x_r - x_e)/n_x; \quad i=0, 1, 2, \dots, n_x$$

$$\phi_0 = V_e$$

note!  $P_0$  is irrelevant -

$$\phi_{n_x} = V_r$$

and will have a surface term to fix boundary cond.

$$\begin{array}{c|ccccc|c|ccccc|c} & -2 & 1 & & & & \phi_1 & & & & P_1 + \frac{\epsilon_0}{\Delta x^2} V_e \\ & 1 & -2 & 1 & & & \phi_2 & & & & P_2 \\ & & 1 & -2 & 1 & & \phi_3 & & & & P_3 \\ & & & \ddots & & & & & & & \vdots \\ & & & & -2 & 1 & & & & & P_{n_x-3} \\ & & & & 1 & -2 & 1 & & & & P_{n_x-2} \\ & & & & & 1 & -2 & \phi_{n_x-1} & & & P_{n_x-1} + \frac{\epsilon_0}{\Delta x^2} V_r \end{array} = -\frac{\Delta x}{\epsilon_0}$$

Matrix has tridiagonal structure and may be rapidly inverted to find the  $\phi_i$ .

- Sparse matrices need not be stored in full (waste of memory)

Many other methods exist to solve the discretized field equations. These methods fall into 3 broad classes:

### 1) Direct Matrix Methods

- Fast inversion of sparse matrices.

### 2) Spectral Methods

- Fast Fourier Transform (FFT)
  - Periodic boundary conditions
  - sine transform [ $g=0$  on grid boundary]
  - FFT + capacity matrix for arbitrary conductors
  - Free space boundary conditions

### 3) Relaxation Methods

- Successive overrelaxation (SOR)
  - General boundary conditions and structures.
- Multigrid (good, fast, <sup>accurate</sup> method for complicated boundaries)

Sometimes these methods may be combined. For example, one might employ spectral methods transversely and invert the tri-diagonal matrix longitudinally.

Other discretization procedures are also widely employed:

- Finite elements
- Variational
- Monte-Carlo .....

The various methods of field solution are central to the efficient numerical solution of intense beams. It is not possible to review them all here. But before discussing particle weighting, we will first overview the important spectral methods and FFTs.

## Spectral Methods and the FFT

The spectral approach combined with numerically efficient Fast Fourier Transforms (FFTs) is commonly used to solve the Maxwell Equations on a discrete spatial grid.

- Approach provides spectral information on fields that can smooth the interactions.

Illustrate in 1-d for simplicity (multidimensions analogous)

$$\frac{d^2\phi}{dx^2} = -\frac{p}{\epsilon_0}$$

Continuous Fourier Transforms: (Reminder)

$\tilde{\phi}(k) = \int_{-\infty}^{\infty} dx e^{ikx} \phi(x)$	$\tilde{p}(k) = \int_{-\infty}^{\infty} dx e^{ikx} p(x)$
$\phi(x) = \int_{-\infty}^{\infty} \frac{dk}{2\pi} e^{-ikx} \tilde{\phi}(k)$	$p(x) = \int_{-\infty}^{\infty} \frac{dk}{2\pi} e^{-ikx} \tilde{p}(k)$

Transform  $\Rightarrow$   $k^2 \tilde{\phi}(k) = \frac{\tilde{p}(k)}{\epsilon_0}$

$$\phi(x) = \int_{-\infty}^{\infty} \frac{dk}{2\pi} e^{-ikx} \frac{\tilde{p}(k)}{\sqrt{\epsilon_0 k^2}}$$

Similar procedures work for the field on a discrete and finite spatial grid.

Discrete Fourier Transform:

$$x_j = x_{\min} + \Delta_x \cdot j' \quad , \quad \Delta x = (x_{\max} - x_{\min}) / N_x \quad ; \quad j' = 0, 1, 2, \dots, N_x$$

$$\phi_j = \phi(x_j)$$

$N_x + 1$  grid points. (take even)

$$k_n = \frac{2\pi n}{(N_x+1)\Delta x} \quad n = -\frac{(N_x+1)}{2}, \dots, \frac{N_x+1}{2}$$

$$\tilde{\phi}(k_n) = \int_{-\infty}^{\infty} dx e^{ik_n x} \phi(x) \approx \Delta x \sum_{j=0}^{N_x} e^{ik_n x_j} \phi_j \\ \approx \Delta x \sum_{j=0}^{N_x} e^{\frac{i2\pi n j}{N_x+1}} \phi_j$$

Discrete Transform:

$$\tilde{\phi}_n \equiv \tilde{\phi}(k_n)$$

$$= \Delta x \sum_{j=0}^{N_x} e^{\frac{i2\pi n j}{N_x+1}} \phi_j$$

Note that  $\tilde{\phi}_n$  is periodic in  $n$  with period  $N_x + 1$

$$\tilde{\phi}_{-n} = \tilde{\phi}_{N_x+1-n}$$

Let  $n$  vary from:  $n = 0, 1, 2, \dots, N_x$

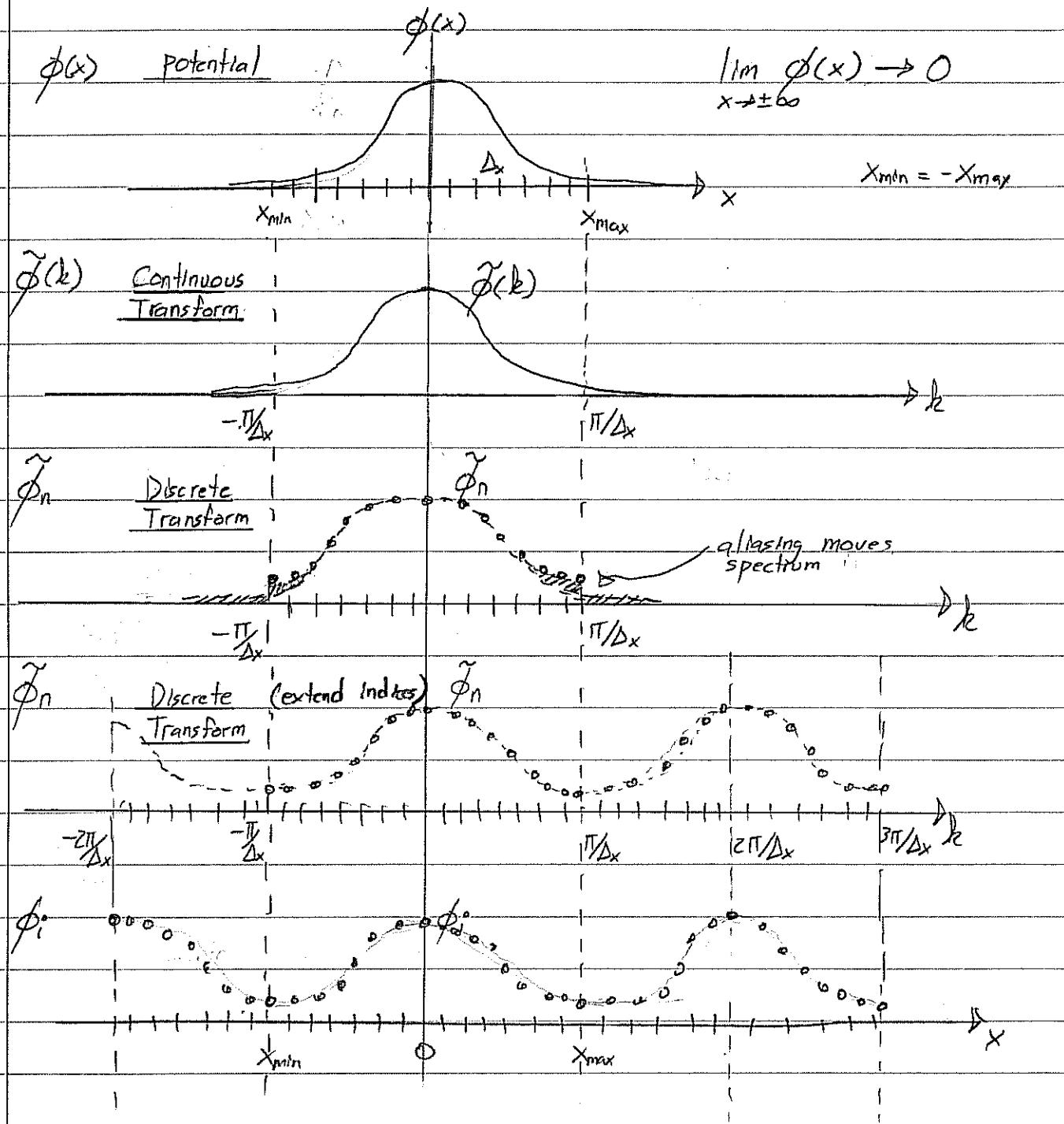
so  $n$  and  $j$  have same ranges.

Then an inverse transform can be exactly constructed!

$$\phi_j = \frac{1}{(N_x+1)\Delta x} \sum_{n=0}^{N_x} e^{-\frac{i2\pi n j}{N_x+1}} \tilde{\phi}_n$$

The discrete transform constructed describes a periodic problem if indices are extended.

- Discretization errors (aliasing) can occur.



Application of Discrete Fourier transforms to solve  
Poisson's Equation:

$$E_x = -\frac{\partial \phi}{\partial x} \Leftrightarrow E_{xj} = -\frac{\phi_{j+1} - \phi_{j-1}}{2\Delta x}$$

$$\frac{d^2\phi}{dx^2} = -\frac{p}{\epsilon_0} \Rightarrow \frac{\phi_{j+1} - 2\phi_j + \phi_{j-1}}{\Delta x^2} = -\frac{p_j}{\epsilon_0}$$

Applying the discrete transform one obtains:

$$\tilde{E}_{xn} = -i K_n \tilde{\phi}_n \quad K_n = k_n \left[ \frac{\sin(k_n \Delta x)}{k_n \Delta x} \right]$$

$$k_n = k_n \text{ dif.}(k_n \Delta x)$$

$$k_n = \frac{2\pi n}{(n_x+1) \Delta x}$$

$$\text{dif}(x) = \frac{\sin x}{x} \quad \text{from diffraction theory}$$

and Poisson's eqn becomes

$$\tilde{\phi}_n = \frac{\tilde{p}_n}{\epsilon_0 K_n^2} \quad K_n^2 = k_n^2 \left[ \frac{\sin(\frac{k_n \Delta x}{z})}{\frac{k_n \Delta x}{z}} \right]^2 \\ = k_n^2 \text{dif}^2(\frac{k_n \Delta x}{z})$$

Note:

- Factors of  $K_n^2$  need only be calculated once per simulation (store values).

Aside - Example Derivation of Discrete Transform Formula.

$$E_x = \frac{1}{2\Delta x} (\phi_{j+1} - \phi_{j-1})$$

$$\left\{ \begin{array}{l} \phi_j = \frac{1}{(n_x+1)\Delta x} \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \tilde{\phi}_n \\ E_x = \frac{1}{(n_x+1)\Delta x} \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \tilde{E}_n \end{array} \right.$$

Substitute transforms in difference formula:

$$2\Delta x \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \tilde{E}_n =$$

$$= - \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \tilde{\phi}_n \left\{ e^{-i\frac{2\pi n j}{n_x+1}} - e^{i\frac{2\pi n j}{n_x+1}} \right\}$$

$$\Delta x \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \tilde{E}_n = i \sum_{n=0}^{n_x} e^{-i\frac{2\pi n j}{n_x+1}} \sin\left(\frac{2\pi n}{n_x+1}\right) \tilde{\phi}_n$$

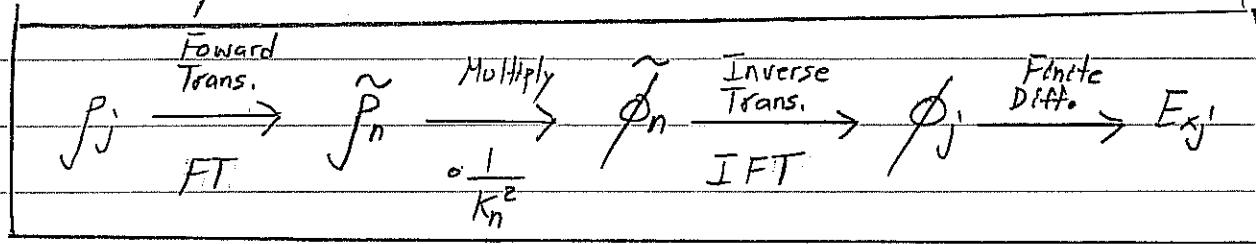
This equation must hold true for each complex term  $e^{-i\frac{2\pi n j}{n_x+1}}$  to be valid for general  $j$ .

$$\Rightarrow \tilde{E}_n = \frac{i}{\Delta x} \sin\left(\frac{2\pi n}{n_x+1}\right) \cdot \tilde{\phi}_n$$

$$\lambda_n = \frac{2\pi n}{(n_x+1)\Delta x}$$

$$\Rightarrow \tilde{E}_n = i \lambda_n \left[ \frac{\sin(\lambda_n \Delta x)}{\lambda_n \Delta x} \right] \tilde{\phi}_n$$

## Typical discrete Fourier Transform Field Solution (not optimized)



The Fast Fourier Transform (FFT) makes this procedure numerically efficient.

- Discrete transform (no optimization)  
 $\sim (N_x+1)^2$  complex operations.
- FFT exploits symmetries to reduce needed operations to  $\sim (N_x+1) \ln(N_x+1)$ ,  
 — Huge savings for large  $N_x$ .
- The needed symmetries exist only for certain numbers of grid points. In the simplest manifestations:

$$N_x + 1 = 2^p \quad p = 1, 2, 3, \dots$$

- reduced gridding freedom,
- other manifestations allow  $2^p$  and products of prime numbers for more possibilities ...

The FFT can be combined with other procedures such as capacity matrices to implement boundary conditions for interior conductors, etc. This allows rapid field solutions in complicated geometries.

FFT is fastest method for simple geometry.

## Particle Weighting

Electrostatic!

We have outlined methods to solve Maxwell's equations on a discrete spatial grid. To complete the description we must:

- Specify how to deposit macro-particle charges and currents on the mesh.
- Specify how to interpolate fields on the spatial grid points to the macroparticle coordinates (not generally on the mesh) to apply in the particle advance.
- Smooth interactions resulting from the small number of macro-particles to reduce artificial collisions resulting from the use of an unphysically small number of macro-particles.

This is called the "particle weighting" problem.

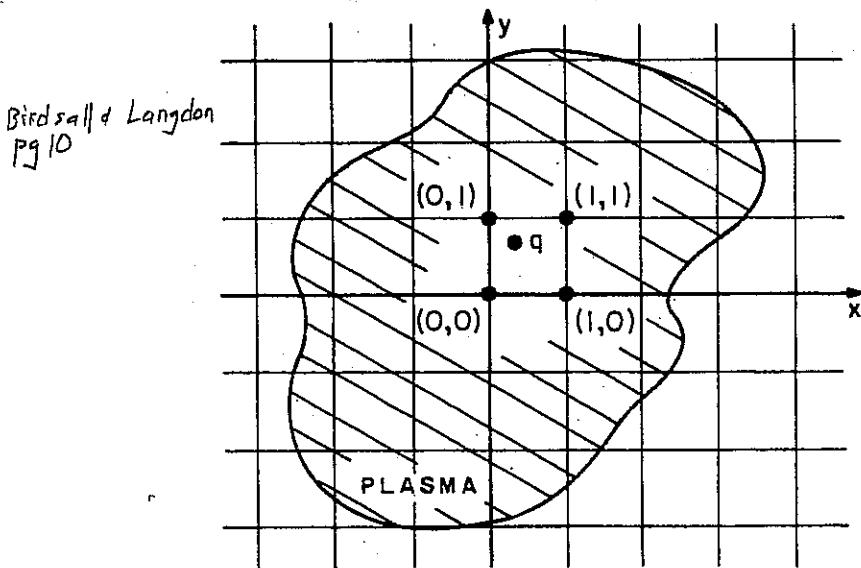


Figure 2-2b A mathematical grid is set into the plasma region in order to measure charge and current densities  $\rho, J$ ; from these we will obtain the electric and magnetic fields  $E, B$  on the grid. A charged particle  $q$  at  $(x, y)$  will typically be counted in terms of  $\rho$  at the nearby grid points  $(0,0), (1,0), (0,1)$  and in terms of  $J$  at the faces between these points. The force on  $q$  will also be obtained from the fields at these nearby points.

If it is found that it is usually better to employ the same weighting schemes to both deposit the Macro-particles' charges and currents on the mesh and to extrapolate the fields at gridded points to the Macro-particles, unphysical

- Avoids self-forces where the particle accelerates itself.

Many methods of particle weighting exist. Can be grouped into 4 categories:

1) Nearest Grid Point

2) Cloud In Cell (CIC)

- Shaped particles. Linear  $\Rightarrow$  Particle-in-Cell PIC

3) Multipole

- Dipole, subtracted dipole, etc.

4) Higher order Methods

- Splines

-  $k$  space cut-offs in discrete transforms

:

Possible hybrid methods also exist. We will illustrate methods 1) and 2) in 1D for electrostatic problems. Generalizations to higher dimensions are straightforward. Descriptions of methods can be found in the literature.

## 1) Nearest Grid Point:

Assign charges to the nearest grid cell.

- Fast and simple

- Noisy

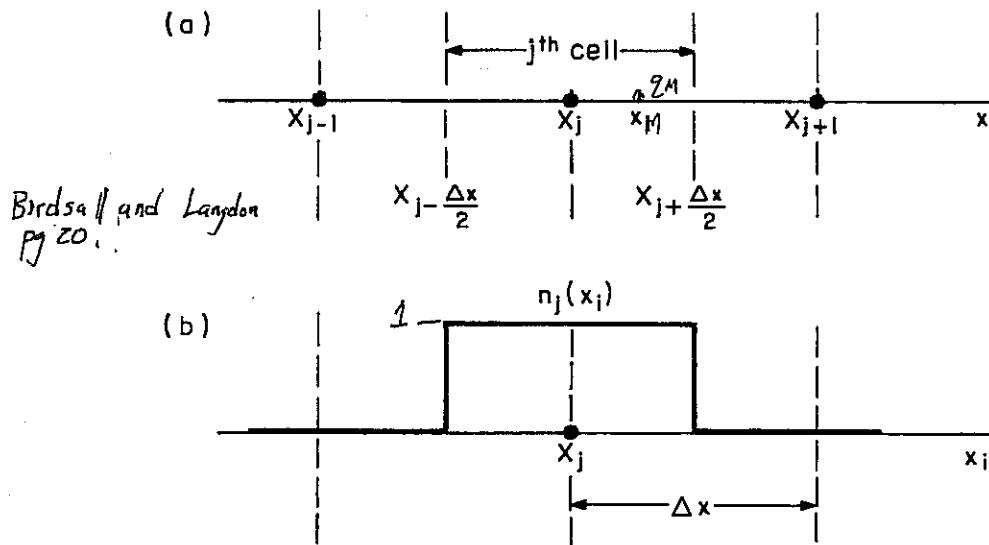


Figure 2-6a (a) Zero-order particle and field weighting, also called nearest-grid-point, or, NGP. Particles in the  $j^{\text{th}}$  cell, that is, with positions  $x_i \in X_j \pm \Delta x/2$ , are assigned to  $X_j$  to obtain grid density  $n_j(X_j)$ . All of these particles are acted on by the field at  $X_j$ ,  $E(X_j)$ . (b) The density  $n_j(X_i)$  at point  $X_i$  due to a particle at  $X_M$  as the particle moves through the cell centered on  $X_j$ . This density may be interpreted as the effective particle shape.

$\nearrow M \rightarrow X_M = \text{charge and coordinate of Macro-particle}$

$X_j = \text{closest grid cell.}$

$$\begin{array}{l} \text{charge : } q_j = q_M \\ \text{Deposition} \end{array}$$

$$\text{Field : } E_x \Big|_{x=X_M} = E_j$$

## 2) Cloud-in-Cell

Shaped macroparticles pass freely through each-other.

- Smoother than Nearest Grid Point.
- For linear interpolation results in simple, commonly used "Particle-In-Cell" (PIC) method.

Illustrate 1D PIC "Area" weighting!

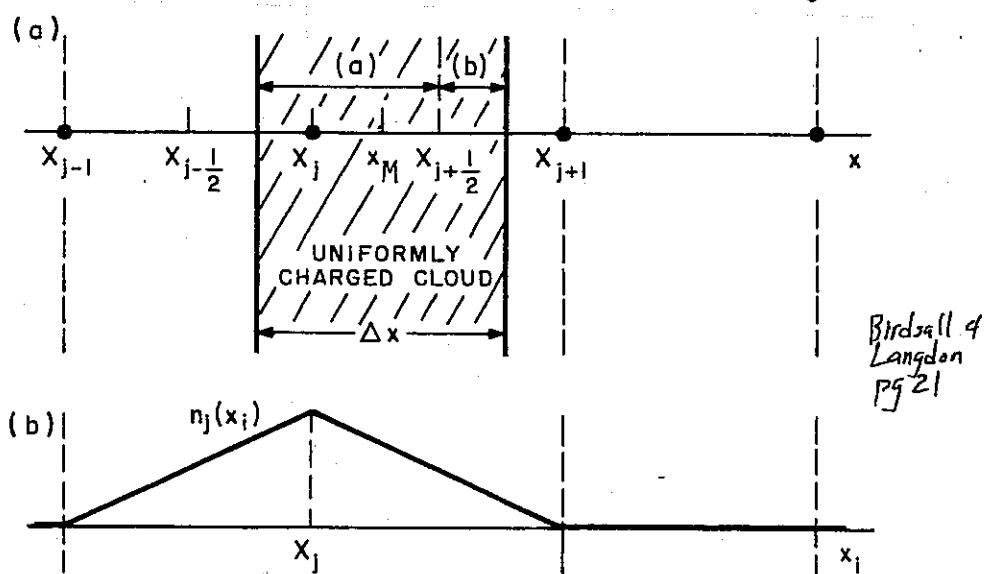


Figure 2-6b (a) First-order particle weighting, or cloud-in-cell model CIC. The nominal finite-size charged particle, or cloud, is one cell wide, with center at  $x_M$ . This weighting puts that part of the cloud which is in the  $j^{\text{th}}$  cell at  $x_j$ , fraction (a), and that part which is in the  $(j+1)^{\text{th}}$  cell at  $x_{j+1}$ , fraction (b). This weighting is the same as applying NGP interpolation to each elemental part. (b) The grid density  $n_j(x_i)$  at point  $x_i$  as the particle moves past  $x_j$ , again displaying the effective particle shape  $S(x)$ .

$$\begin{aligned} q_M, x_M &= \text{charge and coordinate of macro-particle} \\ x_j &= \text{closest grid cell} \\ \text{Charge} \quad \left\{ \begin{array}{l} q_j = q_M \left[ \frac{\Delta x - (x_M - x_j)}{\Delta x} \right] = \frac{x_{j+1} - x_M}{\Delta x} \end{array} \right. \\ \text{Deposition} \quad \left\{ \begin{array}{l} q_{j+1} = q_M \left[ \frac{x_M - x_j}{\Delta x} \right] \end{array} \right. \end{aligned}$$

$$\text{Field: } E_x \Big|_{x=x_M} = \left[ \frac{x_{j+1} - x_M}{\Delta x} \right] E_j + \left[ \frac{x_M - x_j}{\Delta x} \right] E_{j+1}$$

## Computational Cycle for Particle Simulations

We now have (simplified) notions of the parts that make up a particle simulation.

- 1) Particle Moving
- 2) Field Solver on a discrete mesh
- 3) Weighting of particles and fields to and from the mesh.

These are combined in a computational cycle!

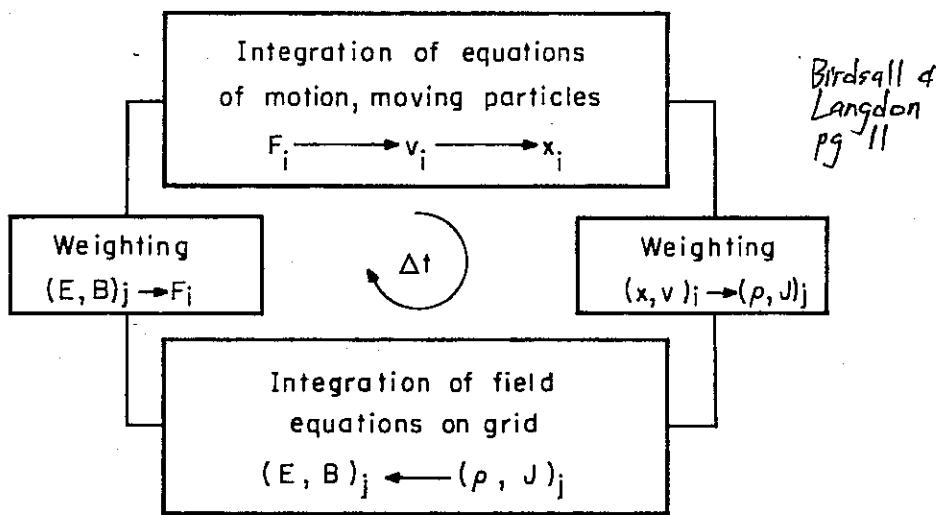


Figure 2-3a A typical cycle, one time step, in a particle simulation program. The particles are numbered  $i = 1, 2, \dots, NP$ ; the grid indices are  $j$ , which become vectors in 2 and 3 dimensions.

At synchronization steps, particle coordinates, velocities, and fields will all be in-phase and diagnostics may be accumulated.

Diagnostics:

Depend on what is analyzed. Typical choices:

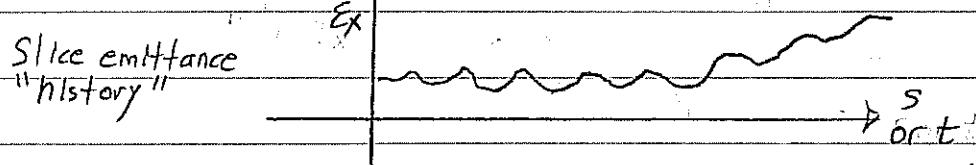
- Moments - statistical sums over the particle distribution (slices and full beam), often plotted as time histories as the beam evolves in the accelerator.

Centroid:  $x_c = \langle x \rangle$

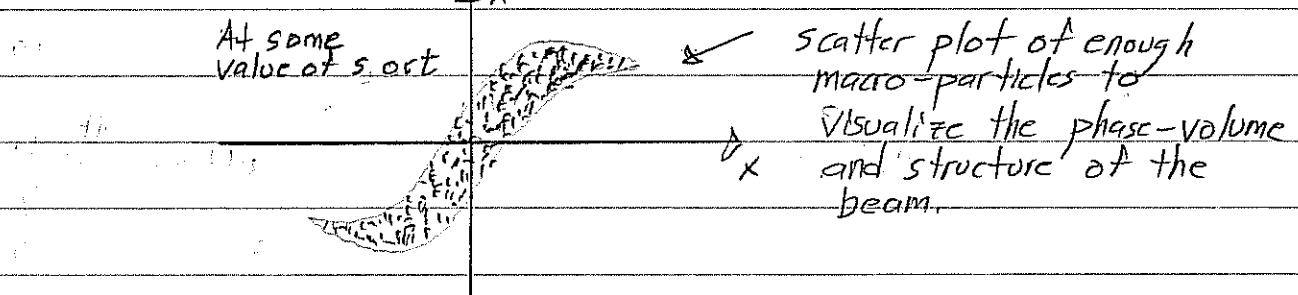
RMS Widths: Beam Size  $\sigma_x = \sqrt{\langle (x-x_c)^2 \rangle}$ , etc.

Emittance:  $\text{Ex} = 16 \left[ \langle (x-x_c)^2 \rangle \langle (x'-x'_c)^2 \rangle - \langle (x-x_c)(x'-x'_c) \rangle^2 \right]^{1/2}$

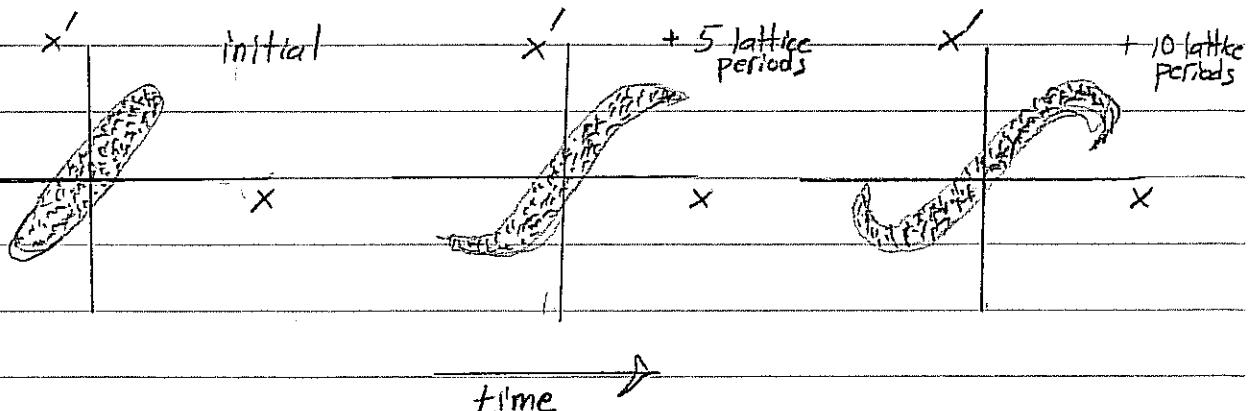
etc. Here:  $\langle \dots \rangle = \frac{1}{N_s} \sum_i^{\text{slice}}$  etc.



- Particle phase-space projections plotted as snapshots in time ( $x-x'$ ,  $y-y'$ ,  $x-y$ ,  $x'-y'$ , etc.).



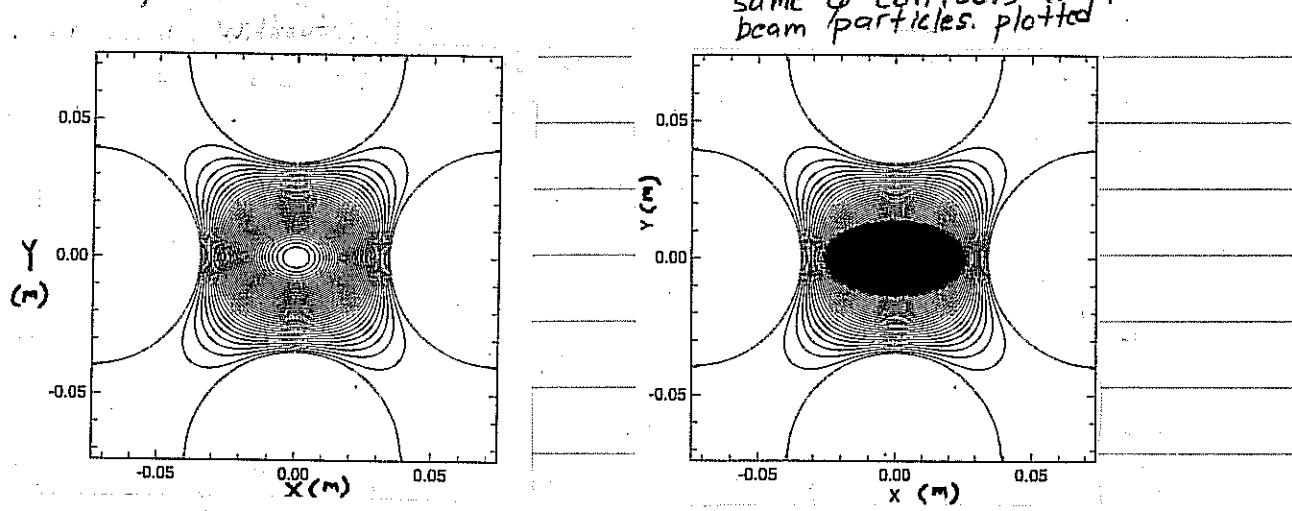
Plotting snapshots at periodic locations allows visualization of the evolution of beam distortions:



• Field Diagnostics:

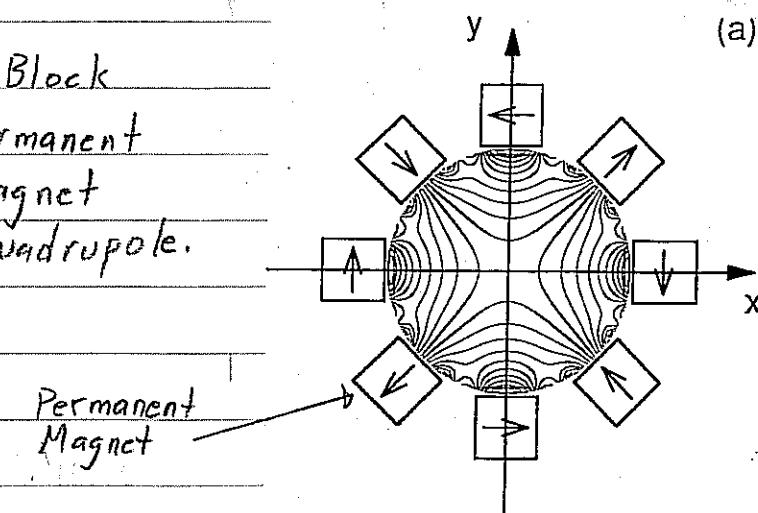
Contours of self and applied fields illustrate field structure.  
Examples:

- 1) Total potential contours of  $\phi$  (self and applied) of an elliptical beam in an electric quadrupole formed by 4 biased rods.



- 2) Applied Field Scalar magnetic potential contours of a permanent magnet lens illustrate the structure of applied field nonlinearities:

$\delta$  Block  
Permanent  
Magnet  
Quadrupole.



(a)

Note:

contours deviate  
from hyperbolic  
near a aperture  
edge.

Numerous other field diagnostics are possible:

- Field energy, multipole moments,

## Particle Loads

To start the simulation, one must specify and "load" the initial distribution function.

In realistic accelerators, focusing elements are  $s$ -varying. In such situations there are "in general" no known equilibrium distribution functions.

Moreover, it is unclear in most cases if the beam is even best thought of as an equilibrium + perturbations as is typical in plasma physics.

Rather, in accelerators, the beam is injected from a source and may only reside in the machine (especially a linac) for a small number of characteristic oscillation periods and may not fully relax to an equilibrium-like state. In such situations, so-called "source-to-target" simulations where the particles are simulated off the source and tracked to the target can be most realistic if carried out with realistic focusing fields, accelerating waveforms, alignment errors, etc.

Unfortunately, such idealized source-to-target simulations can rarely be carried out due to computer limitations.

- Memory limits
- Numerical convergence and accuracy ...

Two ways around this limitation:

- 1) Load experimentally measured distribution at some point in the machine and advance as an initial condition.
- 2) Load an idealized initial distribution.

The 1st option can have practical difficulties:

- Diagnostics often are far from an ideal 6D "snapshot" of the beam phase-space.
  - Much information typically lost.
- Process of measuring the beam can itself change the beam.

Most commonly, some experimental measures such as:

- rms beam sizes  $(x, y) \rightarrow (x', y')$
- rms emittances  $(\epsilon_x, \epsilon_y)$

are loaded in the form of idealized distributions.

It can be insightful to initialize the beam in a simplified manner

- Fewer simultaneous processes can allow one to more clearly see how limits arise.
- Seed perturbations of relevance when analyzing resonance effects, instabilities, halo, etc.

In these situations it is often useful to load a round, continuously focused distribution such as:

- Thermal Equilibrium
- Waterbag
- KV

and then transform the particle coordinates to "match" the local focusing structure of the lattice using a local KV envelope solution:

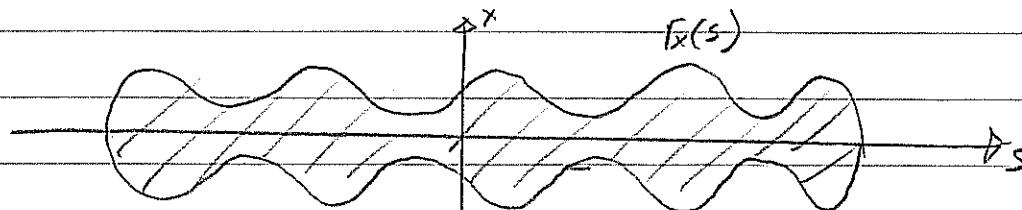
$$\frac{d^2}{ds^2} f_x(s) + R_x(s) f_x(s) - \frac{2Q(s)}{f_x(s) + f_y(s)} - \frac{E_x^2(s)}{f_x^3(s)} = 0$$

$$\frac{d^2}{ds^2} f_y(s) + R_y(s) f_y(s) - \frac{2Q(s)}{f_x(s) + f_y(s)} - \frac{E_y^2(s)}{f_y^3(s)} = 0$$

$R_x(s)$ ,  $R_y(s)$  = lattice focusing constants

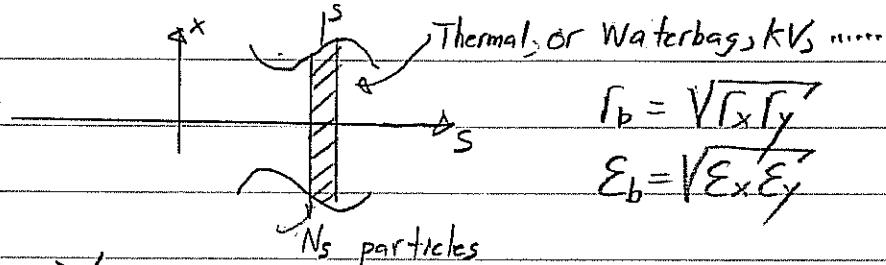
$Q(s)$  = local perveance (specified)

$E_x(s)$ ,  $E_y(s)$  = local emittances (specified)



Procedure:1st

Load in each 1 slice found, continuous distribution:



$$\Rightarrow \vec{x}_i^*, \vec{x}_i^{*'} \text{ specified}$$

2ndTransform the spatial coordinates to match the envelope structure elliptical

$$x_i \rightarrow \frac{r_x}{\Gamma_b} x_i^*$$

$$y_i \rightarrow \frac{r_y}{\Gamma_b} y_i^*$$

3rd

Transform the local thermal velocity spreads to obtain the right average thermal force.

$$x_i^* \rightarrow \frac{\epsilon_x}{\epsilon_{b*}} \frac{\Gamma_b}{r_x} x_i^*$$

$$y_i^* \rightarrow \frac{\epsilon_y}{\epsilon_{b*}} \frac{\Gamma_b}{r_y} y_i^*$$

4th

Add the correct coherent velocity to match the needed envelope angle.

$$x_i^* \rightarrow x_i^* + r_x' \frac{x_i^*}{r_x}$$

$$y_i^* \rightarrow y_i^* + r_y' \frac{y_i^*}{r_y}$$

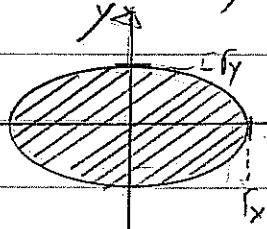
## II Aside - The semi-Gaussian Distribution

It is not necessary to always load something based on a transformation of an equilibrium distribution to get a good gaseous load. Note that for high space-charge intensities:

- Beam space charge will be more or less uniform out to the edge, where the density will rapidly fall to zero.
- If the beam is injected off a uniform temperature source or has relaxed, one expects roughly uniform thermal velocity spread across the cross-section of the beam.

This suggests the so-called "semi-Gaussian" load specified as follows:

- Uniform density within an elliptical beam envelope.



$x_i, y_i$  uniformly distributed for  $(x/x_e)^2 + (y/y_e)^2 \leq 1$ .

- Gaussian distributed thermal velocity spread with the correct coherent velocity to match the needed envelope angles.

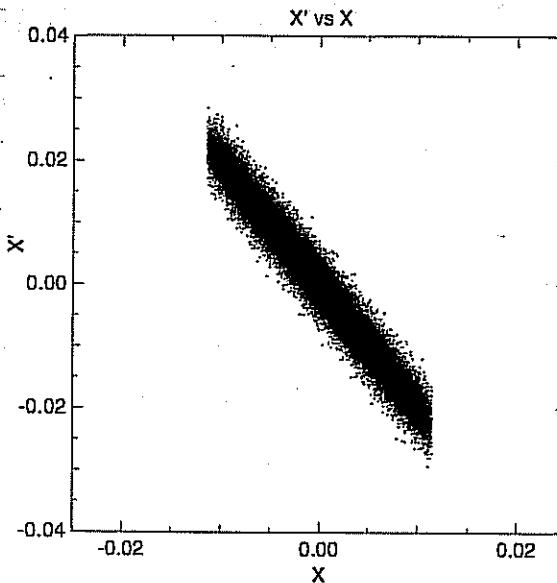
$$x'_p = r_x' \frac{x'_i}{r_x} + \frac{E_x}{2r_x} \tilde{f}_{x_i}$$

$$y'_p = r_y' \frac{y'_i}{r_y} + \frac{E_y}{2r_y} \tilde{f}_{y_i}$$

$\tilde{f}_{x,y}$ , Gaussian

distributed with unit variance ( $\frac{1}{N_s} \sum \tilde{f}_{x,y}^2 = 1$ )

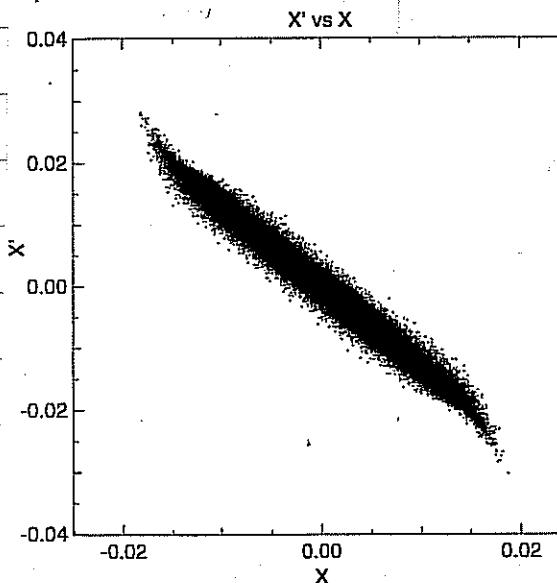
The semi-Gaussian load results in "scared" initial  $x-x'$  and  $y-y'$  phase space projections!



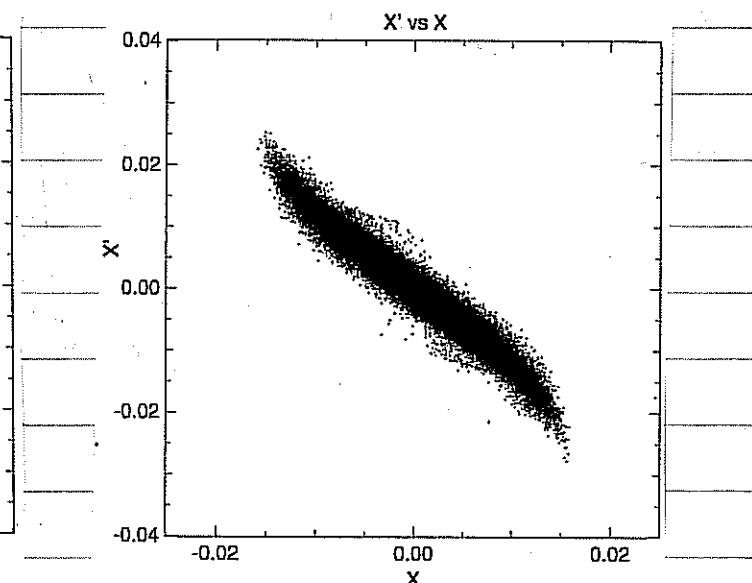
Initial load  
in a symmetric  
FODO AG  
Quadrupole  
transport lattice

The unphysical edges typically relax rapidly without perturbing the low order RMS structure of the beam (envelope match, emittance, etc.).

1 lattice period advance



4 lattice period advance



Numerous other loading techniques exist to address specific issues. //

## Numerical Convergence

Numerical simulations must be checked for proper resolution and statistics to be confident that answers obtained are correct and physical.

### Resolution on discretized quantities

- Time step in particle advance
- Spatial grid for fields

- Statistics (number of macroparticles) to control noise.

More resolution and statistics require more computer time and memory. To be practical, one generally wishes to solve problems with the minimum resources required to achieve correct, converged answers. Unfortunately, there are no set rules on what resolution and statistics are required. This depends on what one is examining, how long the simulations are run, what numerical methods are employed, ... .

### Some general guidance:

- The statistical rms emittances:

$$\epsilon_x = \sqrt{[\langle x^2 \rangle - \langle x \rangle^2]}^{1/2}$$

$$\epsilon_y = \sqrt{[\langle y^2 \rangle - \langle y \rangle^2]}^{1/2}$$

Often prove to be sensitive and easy to interpret measures of numerical differences when plotted as overlaid time histories.

- picks up small phase space distortions induced by numerical errors.

general guidance continued ...

- To get started, find results from similar problems using similar methods.
- Benchmark code and methods against problems with known analytical solutions, established codes using both similar and different numerical methods, ...
- Recheck numerical convergence whenever runs differ significantly or when differing quantities are analyzed.
  - What is adequate for one measure of the beam (say image charge structure) may not be for another (say collective modes).
- Although it is common to increase resolution and statistics till relevant quantities do not vary, it is also useful to purposefully analyze poor resolution and statistics regimes so the characteristics of unphysical numerical errors can be recognized.
- Expect to make many setup, convergence, and debugging runs for each useful series of simulations carried out.

## Specific Numerical Convergence

S. M. Lund 38/

### Comments:

#### Time Resolution.

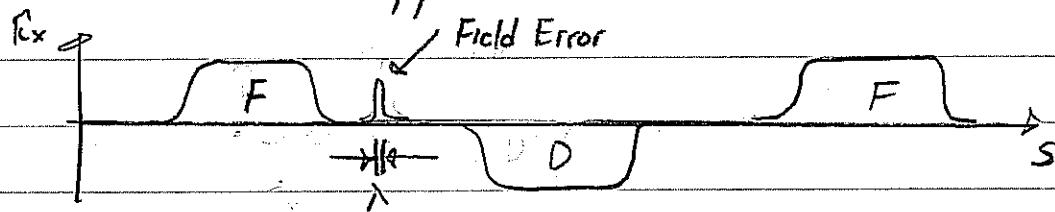
These comments are applicable to both spatial or time steps of the particle advance. We will frame estimates in terms of timesteps.

- Particle coordinates should not move through more than one cell in a single step.  
This is a standard "Courant" condition:

$$\begin{aligned} \Delta x \Delta t &< \Delta x \\ \Delta y \Delta t &< \Delta y \\ \Delta z \Delta t &< \Delta z \end{aligned}$$

for all particles.

- Enough steps should be taken to resolve variations in applied field structures.



$$\Delta s \Delta t \leq \lambda \quad ; \quad \lambda = \text{shortest wavelength of field structures to be modeled}$$

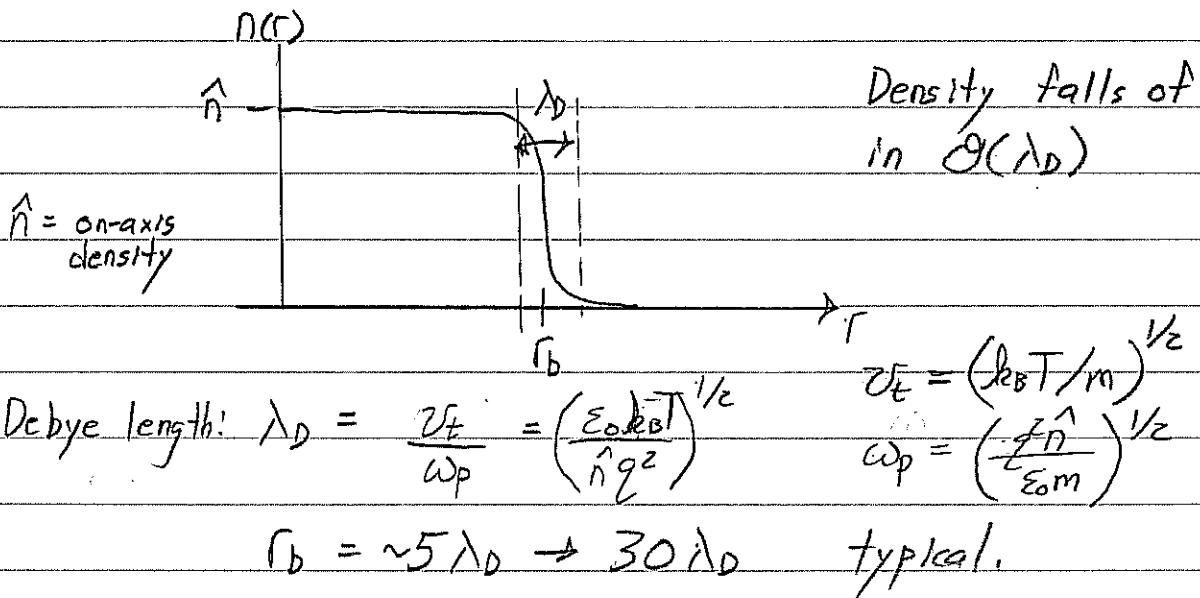
- Phase variations in collective waves (if of interest) should be resolved. For a leap-frog mover this requires:

$\omega = \text{mode frequency component}$

$$\frac{\omega \Delta t}{2\pi} \leq \frac{1}{2}$$

## Spatial Resolution

For cold beams the beam edge can be sharp  
for most reasonable distribution functions:



To resolve edge physics, the mesh should have several cells across the rapid density variation near the edge of the beam.

$$dx, dy < \frac{\lambda_{Dxy}}{2}$$

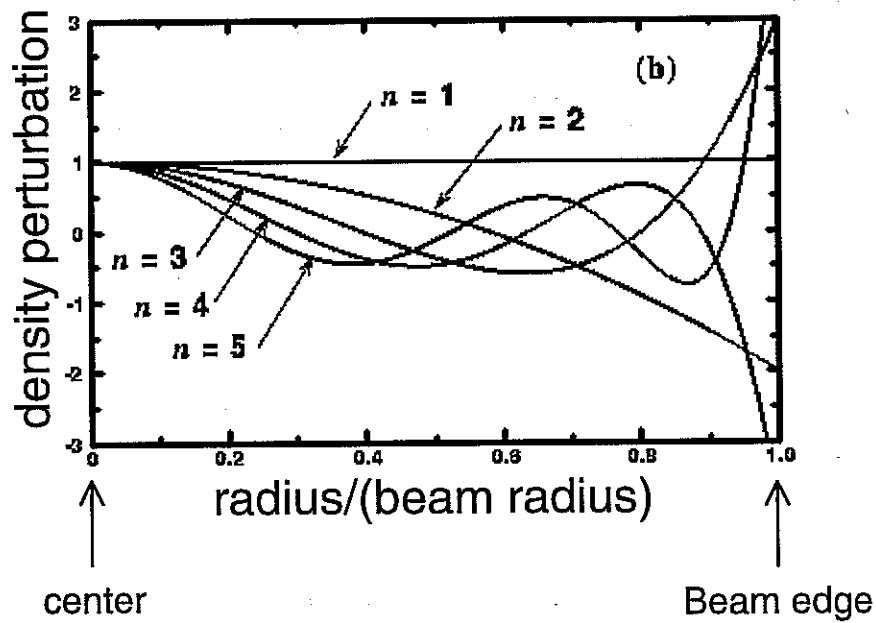
$\lambda_{Dxy} = (k_B T_{xy} / m)^{1/2}$

$T_{xy}$  = local kinetic temperature

The mesh should also resolve relevant spatial scales associated with processes of interest:

- If applied electrostatic fields are calculated from biased conductors, the mesh should resolve conductor structures or special corrections should be made.
- Self-field fluctuations induced by collective modes should be resolved.

Collective mode resolution! From Eigenfunctions to be presented later in the class!



Collective mode density perturbations on a uniform density beam.

$$\Rightarrow dx, dy \ll \lambda \sim \text{shortest characteristic wavelength of modes of interest.}$$

Note that higher order modes ( $n$  larger) will become hard to resolve. Moreover, such perturbations also oscillate rapidly making time ( $s$ ) stepsize resolution likewise difficult.

Statistics.

Collective effects typically require having a significant number of particles  $N_D$  within the characteristic screening radius characterized by the Debye length:

$$2D: N_D = \sum_i \int_{\substack{\text{circle} \\ |\vec{x}| < \lambda_D}} d^2x \delta^{(2)}(\vec{x} - \vec{x}_i) \gg 1$$

$\vec{x}_i$  = macro-particle coordinate.

$$3D: N_D = \sum_i \int_{\substack{\text{sphere} \\ |\vec{x}| < \lambda_D}} d^3x \delta^{(3)}(\vec{x} - \vec{x}_i) \gg 1$$

where:

$$\lambda_D = \frac{2\varepsilon_F}{w_p} = \left( \frac{\varepsilon_0 k_B T}{n e^2} \right)^{1/2}$$

$$w_p = \left( \frac{e^2 n}{\varepsilon_0 m} \right)^{1/2}$$

 $\sum \Rightarrow$  sum over all macro particles.

In simulations of higher order collective modes it may also be necessary to have a significant number of particles per cell on a mesh that resolves the relevant spatial variations of mode induced self-field fluctuations.

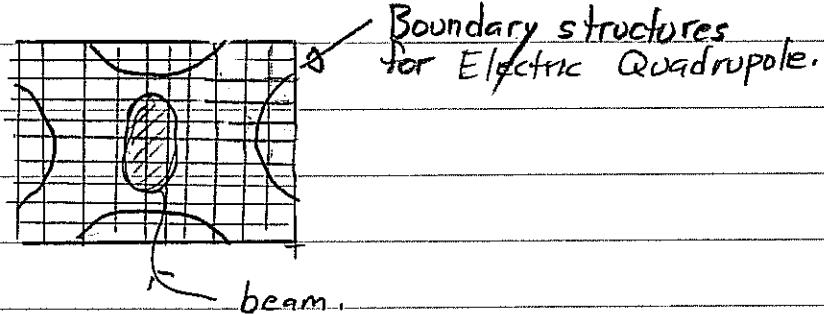
$$2D: N_{cell} = \sum_i \int_{\text{cell}} d^2x \delta^{(2)}(\vec{x} - \vec{x}_i) \gg 1$$

$$3D: N_{cell} = \sum_i \int_{\text{cell}} d^3x \delta^{(3)}(\vec{x} - \vec{x}_i) \gg 1$$

- Larger  $N_{cell}$  prevents local self-fields from being noise dominated.
- Larger  $N_{cell}$  leads to larger  $N_D$  typically  $N_D > N_{cell}$ , since  $\lambda_D$  must be resolved on the grid.

Good statistics are only needed in the beam core with the possible exception of certain beam-halo problems and near the beam edge.

- Most beams will only occupy a fraction of the full grid.



statistics should be evaluated in the cells that the beam occupies rather than average grid measures.

No comprehensive rules exist for how good the statistics must be. Individual problems must be checked and verified. Some general comments!

- What is adequate will typically depend on what is analyzed
  - Image fields may be resolved with few particles
  - Collective waves may take many particles if low noise (interpretable) diagnostic projections are needed)
- Longer runs generally require increased statistics
- Poor statistics result in unphysical collisionality that is often characterized by a linear rise in beam emittances with simulation time.

## Classes of Particle Simulations.

How important is smoothing?

3D Beam:  $N \sim 10^{10} - 10^{14}$  particles typical

Simulations:  $N \lesssim 10^8$  practical  
(modern parallel computers) typical  $10^3 - 10^6$

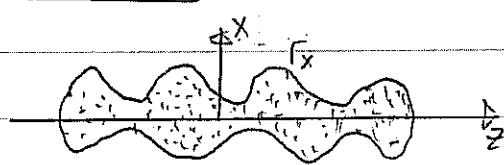
Each simulation particle may represent:  $10^3 \rightarrow 10^{11}$  particles  
in the real beam for 3D simulations.

- Smoothing involved with particle weightings are key to obtaining physical answers and limiting collisionality.

Is the situation really this bad?

- Lower dimensional models typically simulated.

### 3D Model



$N$  point particles with smoothed interactions

### Phase Space:

Physical charge - point charges,  $x, y, z \quad \} \quad 3D$

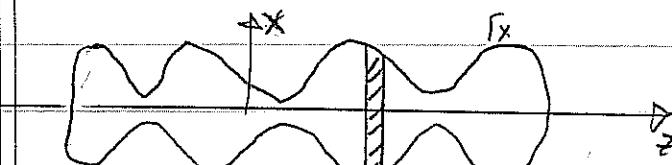
$$\rho = \sum_i q_i \delta(x-x_i) \delta(y-y_i) \delta(z-z_i) \quad p_x, p_y, p_z \quad \}$$

### Smoothed charge

$$\rho = \sum_i q_m f(x-x_i, y-y_i, z-z_i)$$

$q_m$ : Macro particle charge       $f$ : Smoothed shape function

## 2D ⊥ Thin Slice Model

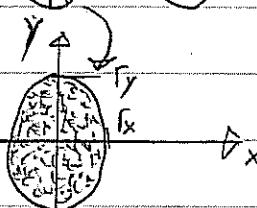


N line charges

with smoothed interactions.

Thin slice

$$\frac{\partial}{\partial z} \ll \nabla_L$$



Phase Space:

$$\begin{aligned} x, y & \} 4D + \text{possible} \\ p_x, p_y & \} 1D \\ p_z & \\ 4D \text{ or } 5D & \end{aligned}$$

"Physical charge" - line charges

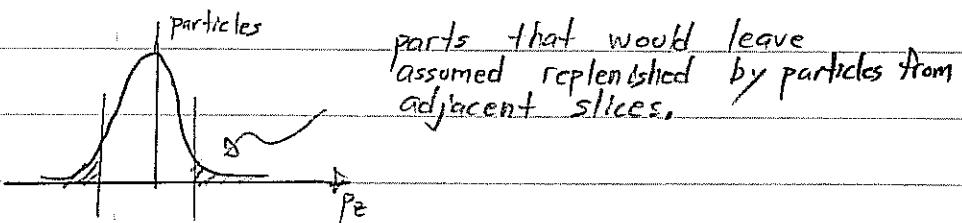
$p = \sum_i \lambda_p \delta(x-x_i) \delta(y-y_i)$

Smoothed charge -  $\lambda_M$  is Macro particle "line charge"

$p = \sum_i \lambda_M f(x-x_i, y-y_i)$   $f$  : smoothing function

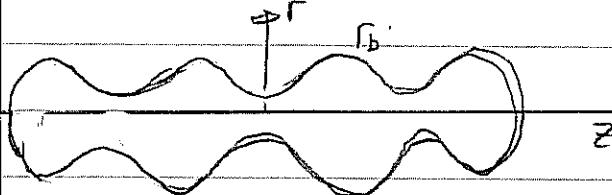
The slice must be tracked in  $s$  with each particle moving the same increment in  $s$  with each step so that a slice maps to a slice.

- If  $p_z$  is included the velocity distribution must be assumed "frozen in".



- Response to acceleration may be modeled with

- "Thick" slice models also possible with periodic boundary conditions on the "slice" to try to recover some 3D effects of a long pulse in a periodic lattice.

2D r-z Model

$N$  charged Rings  
with smoothed interactions

$$\frac{\partial}{\partial \theta} = 0 \quad \text{Axisymmetric}$$

physical charge - cylindrical rings

$$f = \sum_i \frac{Q_i}{2\pi} \frac{\delta(r-r_i)}{r_i} \delta(z-z_i)$$

smoothed charge

$$f = \sum_i \frac{Q_M}{2\pi} \frac{f(r-r_i, z-z_i)}{r_i}$$

Phase Space.

$$(r, z) \rightarrow 4D + \text{possible 1D}$$

$$(p_r, p_z) \rightarrow$$

$$p_r$$

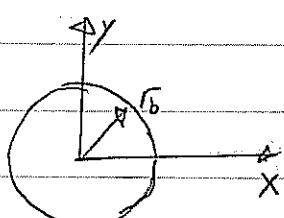
(angular mom.)

4D or 5D.

$Q_M$ : Macro particle charge  
 $f$ : Smoothing function.

- Used to model solenoidal transport of an initial axisymmetric beam.
- Sometimes used to model AG beams with an approximately equivalent,  $s$ -dependant focusing force.

$$\vec{x}_L'' = k_{p0}(s) \vec{x}_L + \dots$$

1D Axisymmetric Model

$N$  charged cylinders with smoothed interactions

$$\frac{\partial}{\partial \theta} = 0$$

Phase - Space

$$(r) \rightarrow 2D + \text{possible 1D}$$

$$(p_r) \rightarrow p_r, p_z$$

2D to 4D

"physical" charge - cylindrical sheets

$$f = \sum_i \frac{Q_i}{2\pi} \frac{\delta(r-r_i)}{r_i}$$

smoothed charge

$$f = \sum_i \frac{Q_M}{2\pi} \frac{f(r-r_i)}{r_i}$$

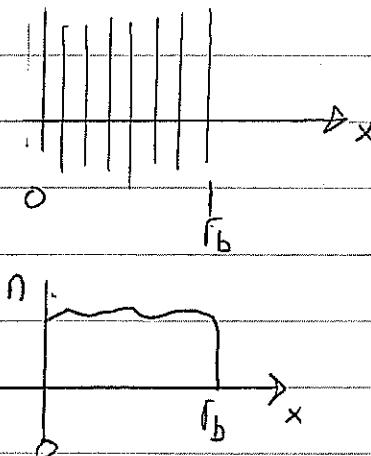
$$Q_M : \text{Macro charge}$$

$$f : \text{Smoothing function}$$

- Simple model for continuously focused, axisymmetric beams

1D slab Model

$N$  charged slabs with smoothed interactions.

Phase-Space

$$\begin{aligned} x &\} 2D + \text{possible} \\ p_x &\} p_y, p_z \end{aligned}$$

$\xrightarrow{\quad}$

2D to 4D

"physical" charge - sheets

$$\rho = \sum_i \sigma_i \delta(x - x_i)$$

smoothed charge

$$\rho = \sum_i \bar{\sigma}_i f(x - x_i)$$

$\bar{\sigma}_i$ : macro particle charge  
 $f$ : smoothing function

- Most simple models, but slab geometry is least physical.

It is not immediately clear how such different models can in many cases represent qualitatively similar collective interactions since force laws can change form with dimension. For example, in free space, we find that:

Model	Free Space: Field due to $i$ th "particle"
3D	$\vec{E} = \frac{q_i (\vec{x} - \vec{x}_i)}{4\pi\epsilon_0  \vec{x} - \vec{x}_i ^3}$
2D	$\vec{E} = \frac{\lambda_i (\vec{x} - \vec{x}_i)}{2\pi\epsilon_0  \vec{x} - \vec{x}_i ^2}$
1D	$E_x = \frac{\sigma_i (x - x_i)}{2\epsilon_0  x - x_i }$

$\sigma_i$  = sheet charge "particle"

The reason these radically different interactions can give similar physics is that the screening associated with collective interactions is found to be similar:

- Debye screening has similar characteristics in each dimension.

Showed 2D

form, in  
W/H Show Mn  
that the  
3D

Scaling obtains  
the same  
Debye length  
definition.

$$\lambda_D = \frac{2\sigma_{\text{thermal}}}{\omega_p} = \left( \frac{\epsilon_0 k_B T}{m g^2} \right)^{1/2}$$

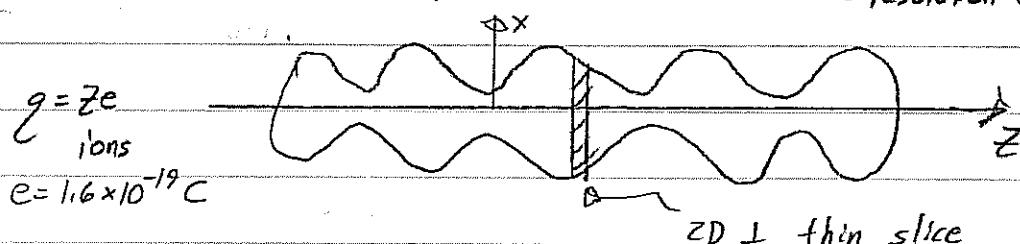
$$\sigma_f = \left( \frac{k_B T}{m} \right)^{1/2}$$

- It is much easier to have a significant number of particles within the characteristic screening distances for lower dimensional problems.

- Lower dimensional simulations can

more easily resolve collective effects.

(Sometimes people run 3D simulations for collective modes and present garbage answers due to resolution difficulties)



$$\lambda \sim 10^{-13} \rightarrow 10^{-2} \text{ cm} \quad \text{typical for intense beams}$$

$$\# \text{ particles} = \frac{\lambda}{cm} \sim \frac{10^4}{Z} \rightarrow \frac{10^{10}}{Z}$$

$$g = \frac{Ze}{q}$$

charge state

- Smoothing still important in lower dimensions and real beam is 3D

## Practical Considerations

### Practical Consideration 1:

#### Computer Memory

Computer memory dictates how large a problem can be run.

- If the problem will not fit into fast memory (RAM), computer performance will be severely compromised.
  - writes to hard disks are slow.

3 main contributions to memory:

- 1) Particle Phase space coordinates
  - 2) Gridded field
  - 3) General code overhead.
- } Dominate for  
large problems

Generally, particle and field quantities are stored in double precision:

Representation	digits (Floating Point)	Bytes memory
Single Precision	8	4
Double Precision	16	8 ←

Relevant case for most problems

#### 1) Particle Phase Space Coordinates

$$N_p = \# \text{ macro particles}$$

$D = \text{Dimension of variables characterizing particles.}$

$$\boxed{\text{Memory} = 8 \times N_p \times D \text{ Bytes}}$$

The dimension  $D$  depends on the specific type of simulation!

// Examples:

$$\underline{3D} \quad D = 7$$

$x, y, z$  for opt.

$$P_x, P_y, P_z, \gamma^{-1}$$

$$\underline{2D} \quad D = 4$$

↓ slice  
in S

$$x, y$$

$$P_x, P_y$$

//

## 2) Gridded Field

For electrostatic problems typically  
need only store  $\phi$  and  $p$  on  
the mesh.

- Discrete Fourier Transform complex,  
but transform is of real functions.  
Proper optimization can be carried out  
using only  $\phi$  and  $p$  arrays.
- Electric field is typically not stored  
and is calculated for each particle  
only where needed.
  - Some methods do store gridded  $E$   
to optimize specific problems / computers.

$$N_m = \# \text{ mesh points.}$$

$$= (n_x + 1)(n_y + 1) \quad \text{2D + slice}$$

$$= (n_x + 1)(n_y + 1)(n_z + 1) \quad \text{3D}$$

$$\begin{matrix} \text{Bytes} \\ | \\ 8 \cdot 2 \end{matrix} \quad \begin{matrix} \phi \\ \rightarrow p \end{matrix}$$

$$\boxed{\text{Memory} = 16 * N_m \text{ Bytes}}$$

Dimensionality is also critical in field solves.

One typically finds:

1D Simulations:

Fieldsolve trivial - only a small fraction of the  
work involved in moving particles.

- Greens' functions can be used.  
(Gauss' Law)

## 2D Simulations

Fieldsolve is typically a small fraction of the work involved in moving particles if fast, gridded methods are used. (e.g., FFT). Special boundary conditions may increase this fraction.

### Numerical Work

Small Fraction particle moving

### Method

FFT with periodic B.C.

FFT with Capacity Matrix

SOR

Dominates particle moving

Greens Function

## 3D Simulations

Fieldsolve typically is comparable to or dominates the work involved in moving particles even if fast, gridded methods are employed.

- Fieldsolve efficiency of critical importance in 3D to optimize run time,
- You could teach whole classes just on 3D electrostatic field solve methods for Poisson's equation.

## 3) General Code Overhead

Some system memory is also used for:

- Scratch arrays for various numerical methods in fieldsolvers, movers, etc.
- Diagnostic routines, graphics packages, etc.

Memory = M <sub>overhead</sub> ~ characteristic of given code Typical 1 MB - 20 MB
---

Some modern graphics packages are large!

## Total memory usage (Electrostatic PIC)

$$\text{Memory} = 8 \times N_p \times D + 16 N_m + \text{Overhead Bytes}$$

Machine RAM capacity should not be exceeded.

### Speed of Computational Cycle:

The speed of the computational advance cycle is also extremely important and dictates the time that runs will take.

- Parts of the code that more time is spent in should be more carefully optimized to minimize operations
  - Particle mover
  - Field solver
  - Frequent diagnostics such as moments.
- Infrequent diagnostics, loaders, problem setup routines etc. often can be coded with less care since they are only executed infrequently. However, diagnostics often take much wall clock time to code!

If is often useful to benchmark the code in terms of

$t_{\text{step}}$  = Time for a computational cycle in an "ordinary" run step.

## Machine Architectures.

Problems may be simulated on :

1) Serial machines

- sometimes small clusters of processors without specific need for parallel programming.

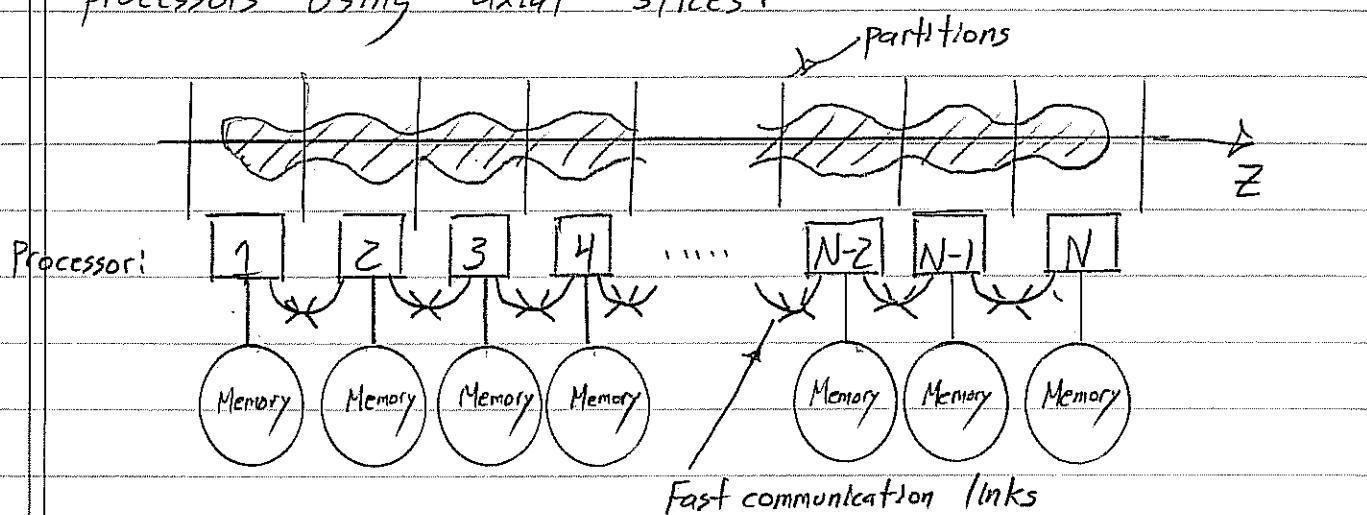
2) Parallel machines

- multi-processor, possibly shared memory but more often linked memory via fast message processing.

Parallel machines have been recently improved with libraries that allow "natural" problem formulation with less effort and they promise to enable much larger simulations. (resolution+statistics)

- Several 100 Million particle simulations possible.

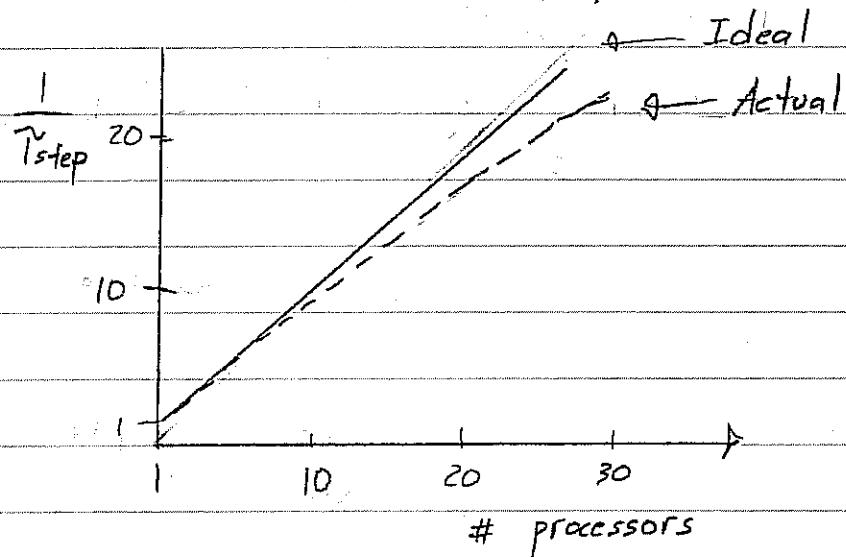
Beams problems may often be conveniently partitioned among processors using axial slices:



- Sharing of data at boundaries is needed for field solves
- Some problems will require the partitions to be adjusted (particle sorting) to maintain the load balance.
  - Processors should ideally share load equally since the slowest will limit the advance.

Ideal parallelization will result in a linear speedup with processor number.

$\bar{T}_{\text{step}} = \text{time per "ordinary" step of computational cycle.}$



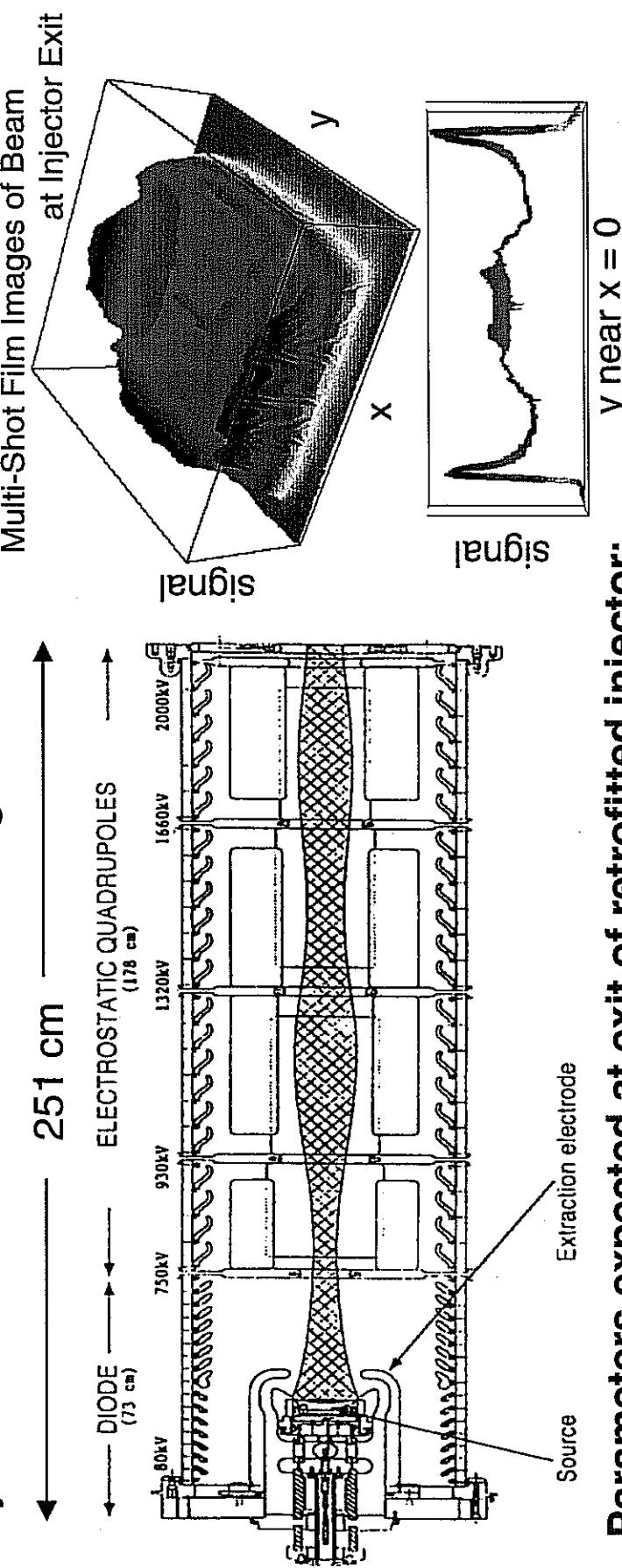
- Actual speedup will be less than ideal due to:
  - overhead in data transfers
  - less than ideal load balance among the processors, resorting of particles, etc.

Even with the significant advances in problem size and speed promised by parallel computers, the solution of realistic 3D beam problems with direct (not gridded) fields will remain far too large a problem to simulate in the foreseeable future. Thus we will always try to exploit computer resources to the maximum extent possible

- Better numerical methods
- Parallelization

# 3D PIC simulations are being used to guide retrofits of an existing ESQ injector at LBL

1st principles, mid-pulse 3D simulations have been carried out to guide injector retrofits aimed at decreasing beam aberrations



Parameters expected at exit of retrofitted injector:

Energy:  $E = 1.71 \text{ MeV}$

Current:  $I = 692 \text{ mA, K}$

Emittance:  $\varepsilon_n = 1.1 \pi \text{ mm-mrad}$

rms edge measure  
measure eliminating empty entrained space

Envelope:  $r_x = 56.3 \text{ mm}$

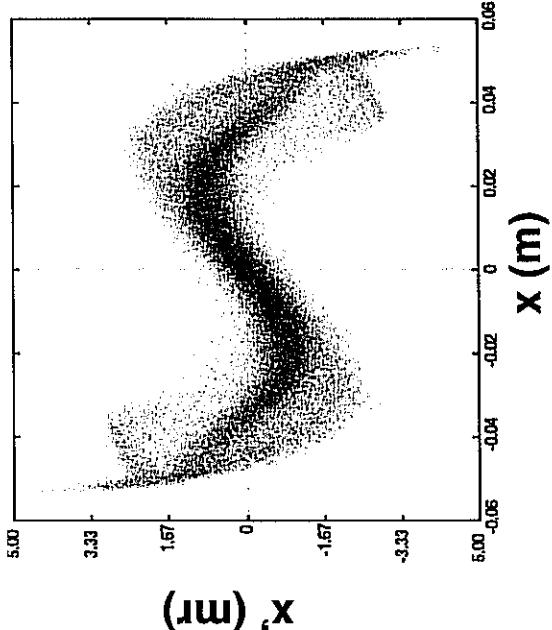
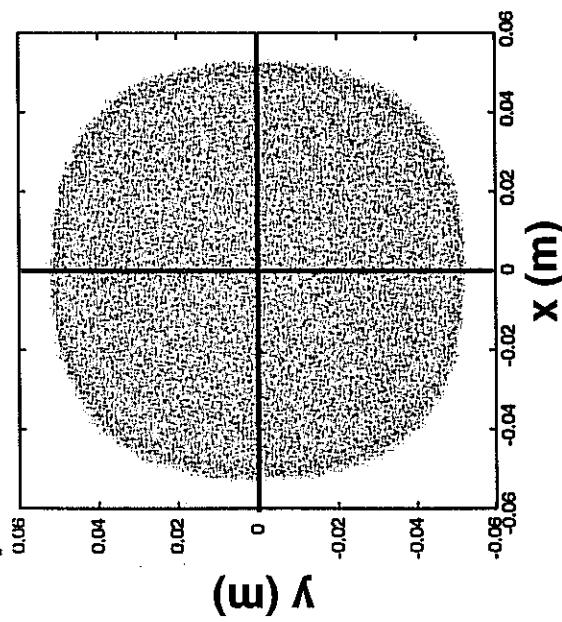
$r_x' = 53.8 \text{ mrad}$

$r_y = 55.7 \text{ mm}$

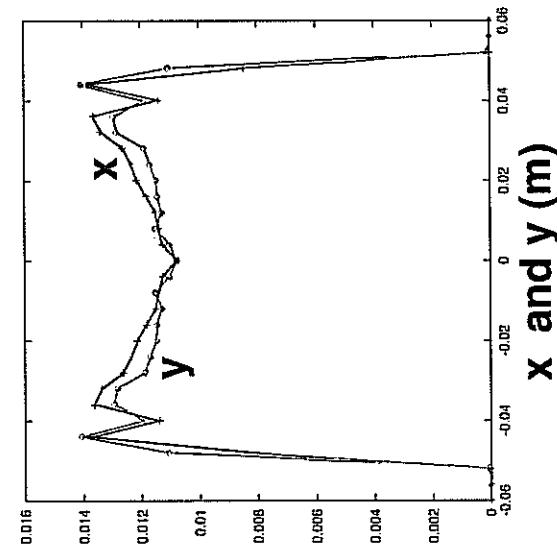
$r_y' = -44.8 \text{ mrad}$

# HCX Injector: Simulations show distribution distortions in the retrofitted injector should be more modest

Mid-pulse distribution projections at exit plane of injector



Density (arb units)



6

Overall convergence and divergence angles removed to illustrate distortions

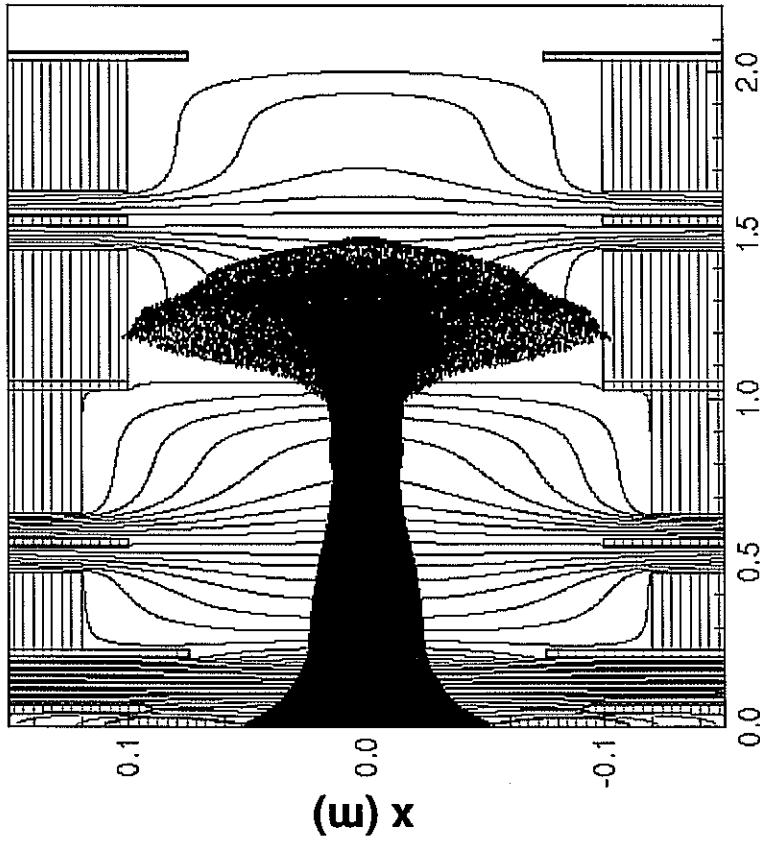


The Heavy Ion Fusion Virtual National Laboratory

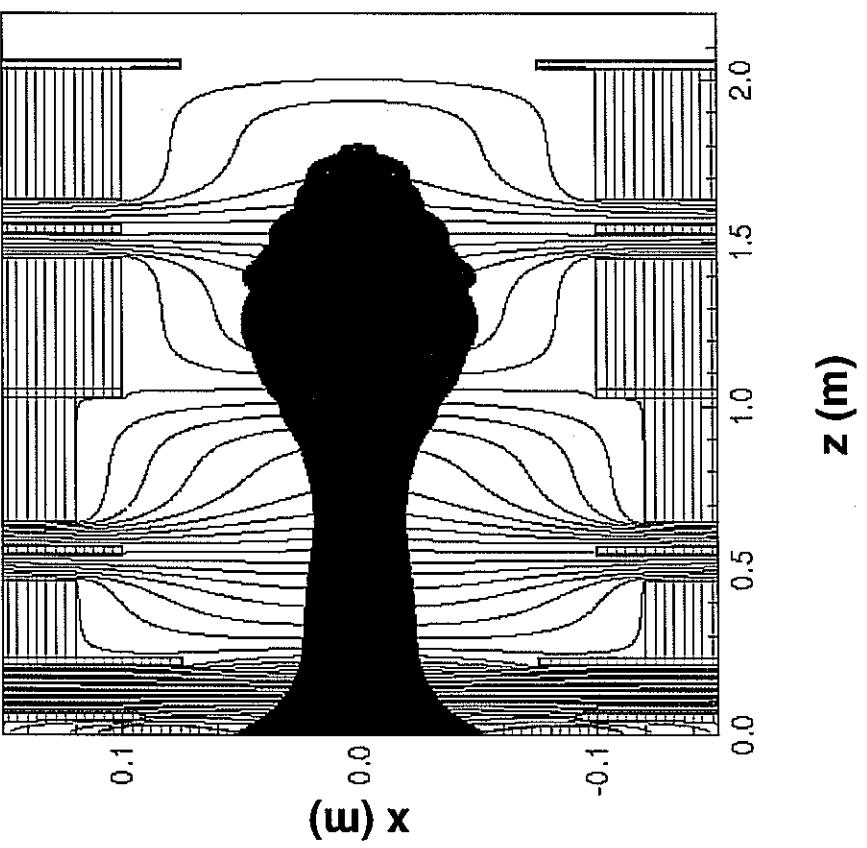
# Simulations show how waveform rise-time determines beam head mismatch

Tuned 1d diode voltage waveform rise-time is 400 ns -- deviations from this lead to significant mismatch effects in the beam head and particle loss with resultant worries about breakdown, electron effects, etc.

Rise-time  $\tau = 800$  ns  
beam head particle loss  $< 0.1\%$



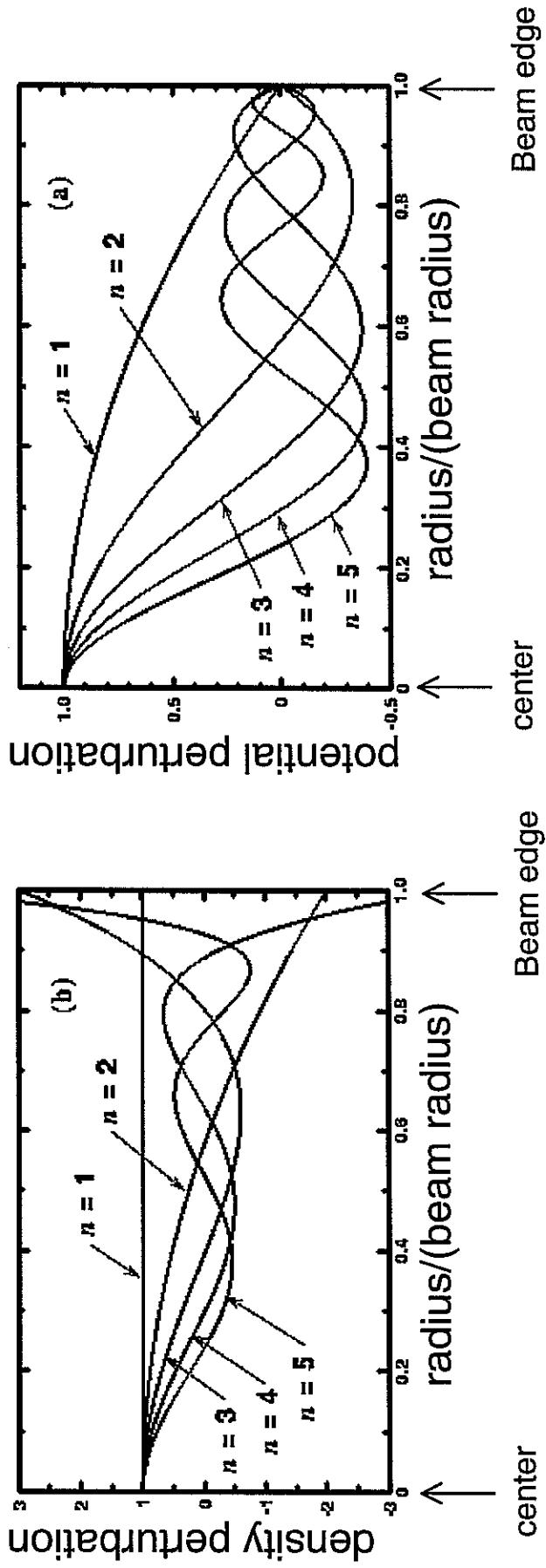
Rise-time  $\tau = 400$  ns  
zero beam head particle loss



# Initial distribution distortions will launch a spectrum of collective mode perturbations that evolve

Kinetic and fluid theories have been employed to analyze perturbations on a uniform density intense-beam equilibrium [Lund and Davidson, Phys. Plasmas, 5 3028 (1998)]

Small Amplitude Perturbations (arbitrary units, kinetic and fluid theory)



Mode Dispersion Relation (fast branch, from fluid theory)

$$\frac{\sigma_n}{\sigma_0} = \sqrt{2 + 2\left(\frac{\sigma/\sigma_0}{\sigma/\sigma_0}\right)^2 (2n^2 - 1)}$$

$\sigma_n$  = mode phase advance  
 $n = 1, 2, 3, \dots$

S.M. Lund 57/

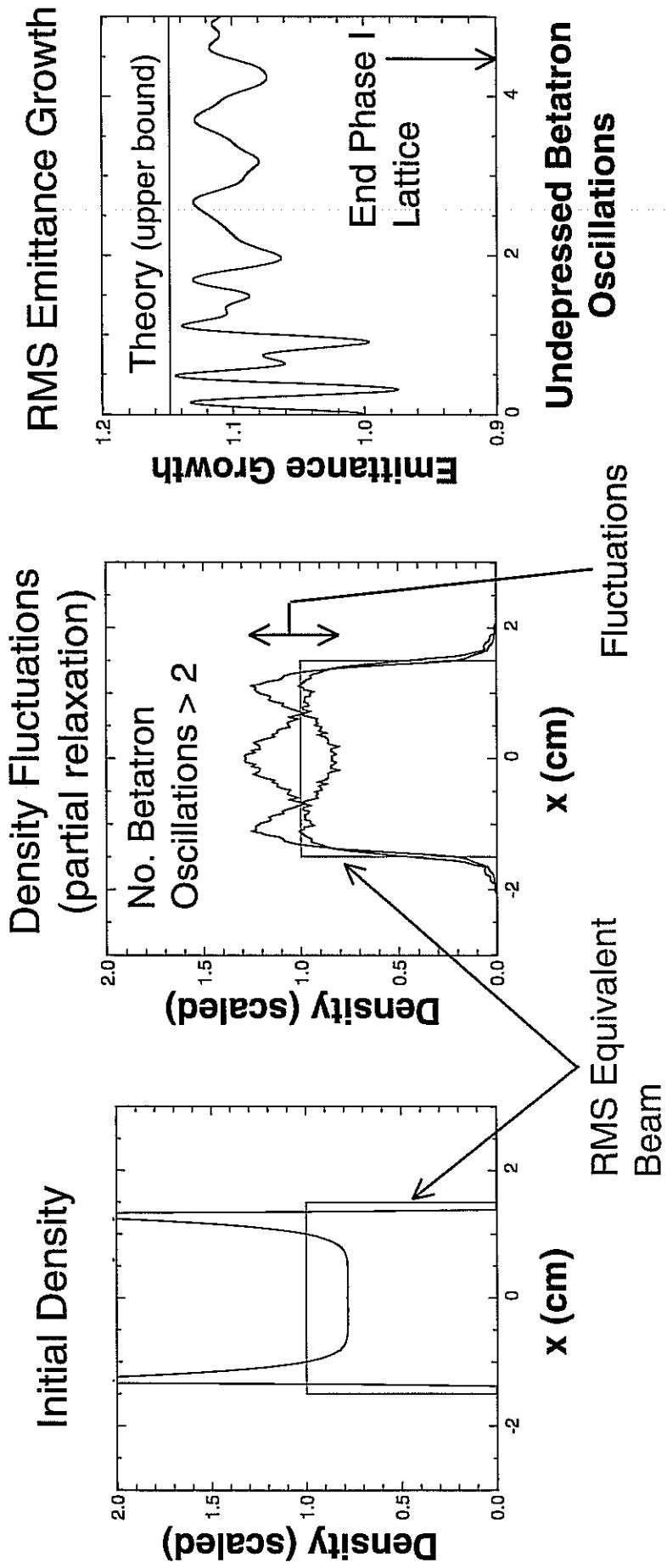
Example:  
 $\sigma_0 = 80^\circ, \sigma/\sigma_0 = 0.2$   
 $\sigma_1 = 115^\circ, \sigma_5 = 182^\circ, \dots$



Perturbations launched by initial distribution nonuniformities can phase-mix to a more uniform profile with increased emittance

Mode spectrum launched can undergo a rapid cascade, settling to a smaller amplitude and lower order distortion

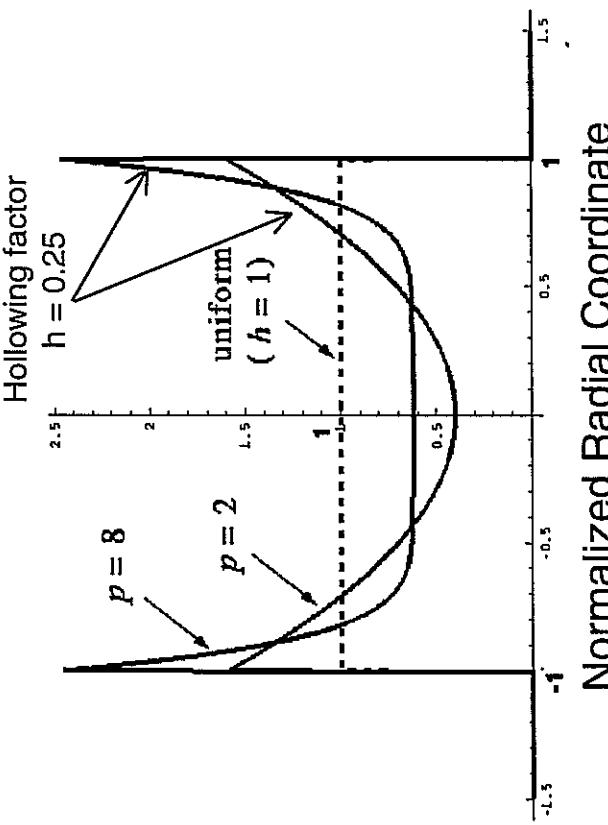
- Approximate conservation constraints employed to bound emittance increases resulting from full relaxation to a uniform profile [Lund, Lee, and Barnard, Proc. Linac 2000, pg. 290]
- How will such evolutions influence the range and interpretation of measurements



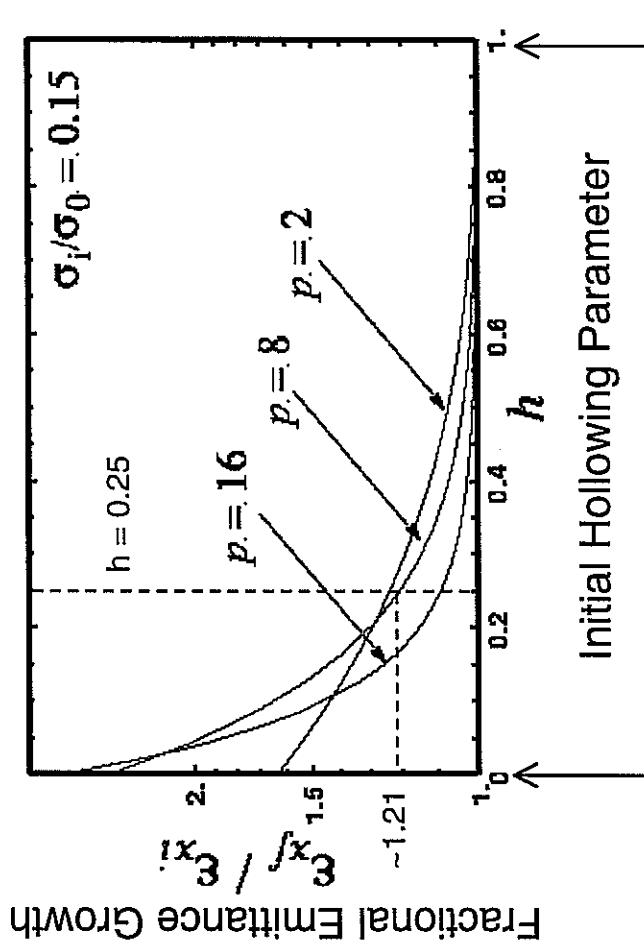
## Analytic theory has been used to parametrically bound emittance growth due to the relaxation of space-charge nonuniformities

Approximate conservation constraints can be employed to estimate maximal emittance increases resulting from the relaxation of an initial nonuniform density profile to a final, uniform profile [Lund, Lee, and Barnard, Proceedings Linac 2000, Monterey, CA, pg. 290]

Initial Density



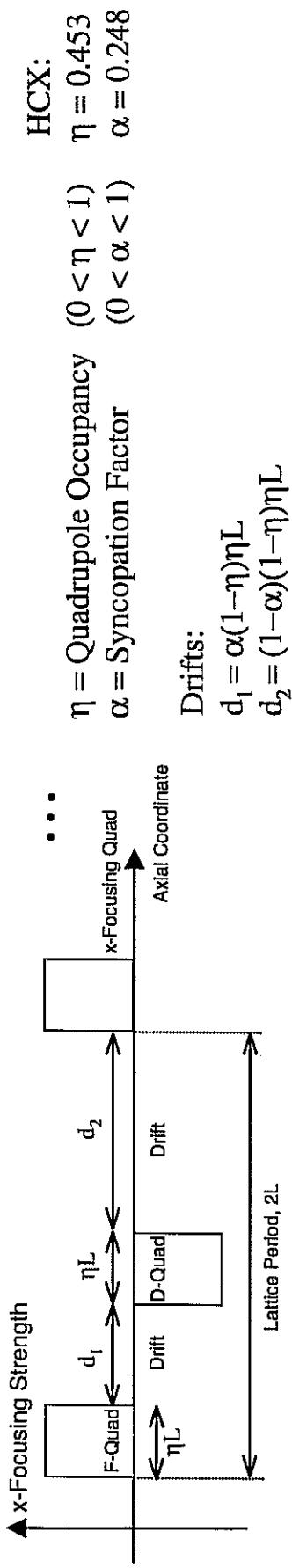
hollowing  $\sim r^p$   
 $h = \text{ratio min to max density}$



Extremely Hollowed      Extremely Hollowed  
 Uniform

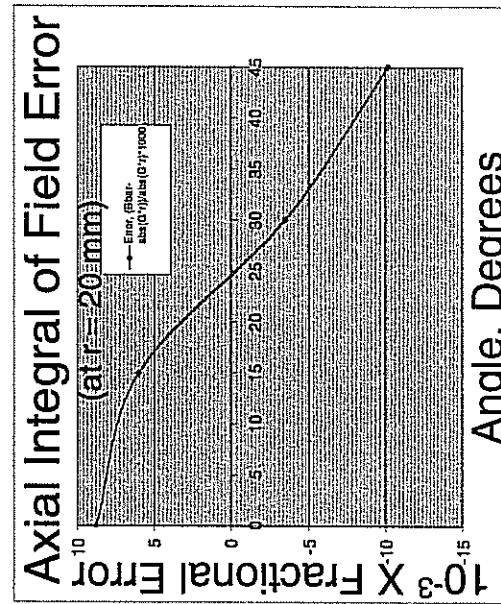
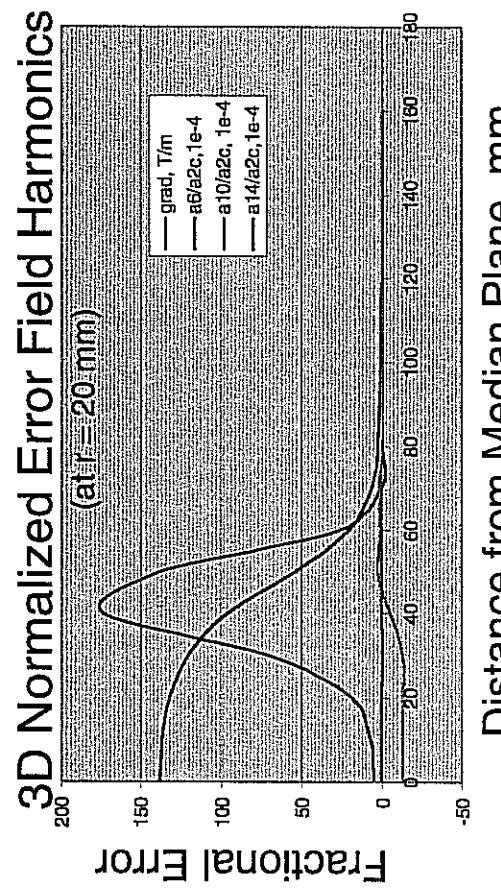
# HCX Phase II: A wide range of parametric simulations have been carried out with realistic magnetic transport lattices

A syncopated magnetic transport lattice of ~ 50 lattice periods has been designed



Superconducting magnetic quadrupoles have been designed and prototyped

$$\begin{aligned} r_p &= 29.5 \text{ mm} & B_q' &= 104 \text{ T/m (max)} \\ l &= 136 \text{ mm} & l_{\text{eff}} &= 101 \text{ mm} \end{aligned}$$



The Heavy Ion Fusion Virtual National Laboratory



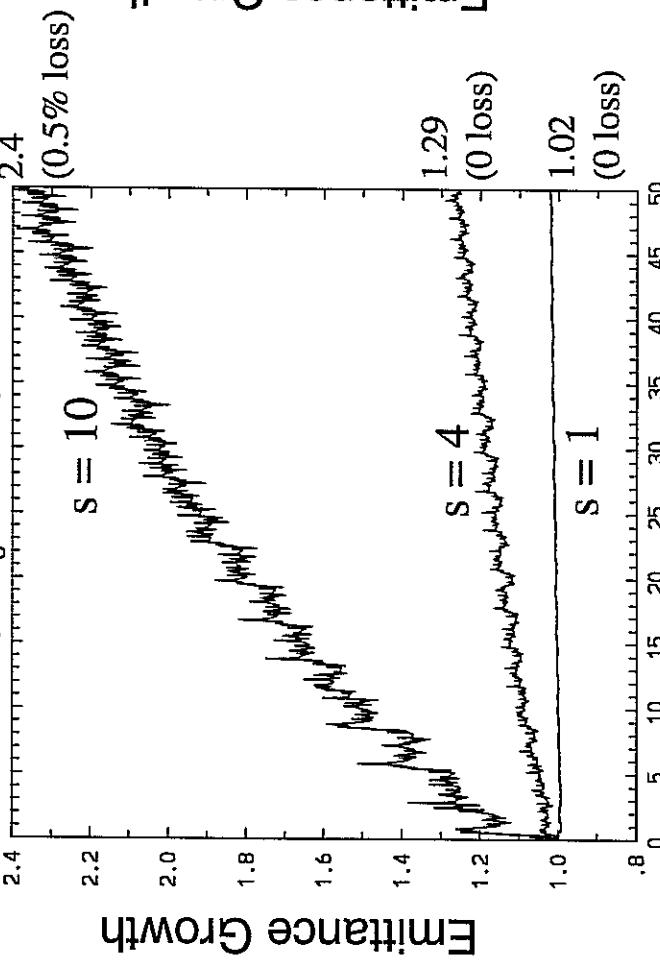
**HCX Phase II:** Processes influencing beam quality and control have been explored --- Example: Nonlinear applied fields and beam quality

Full 3D magnetic field is resolved as:

$$\vec{B} = \vec{B}_{\text{quad}} + \delta\vec{B}$$

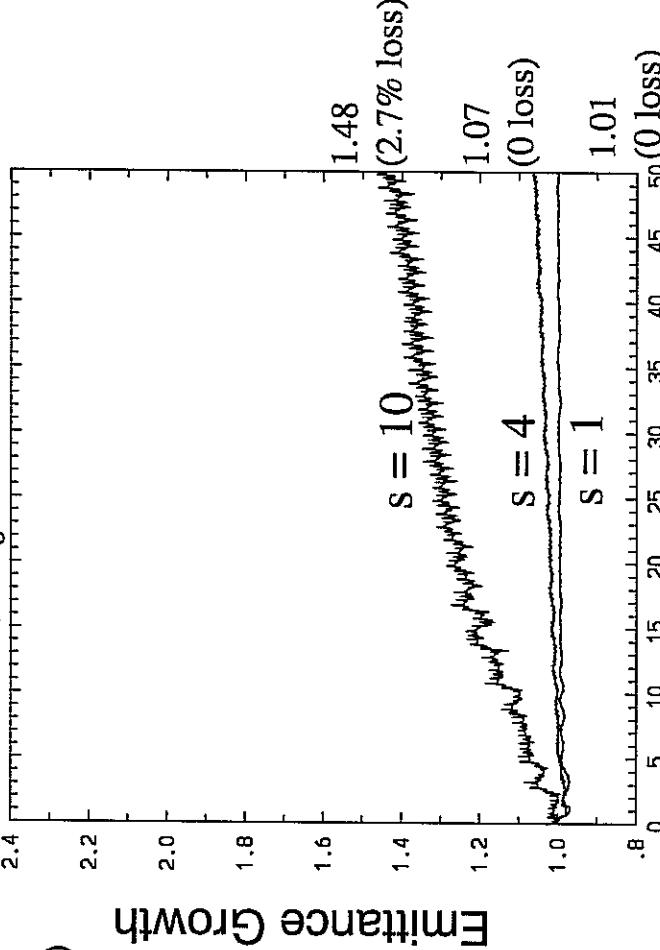
Very Strong Space Charge

( $\sigma/\sigma_0 = 0.086$ )



Intermediate Space Charge

( $\sigma/\sigma_0 = 0.248$ )

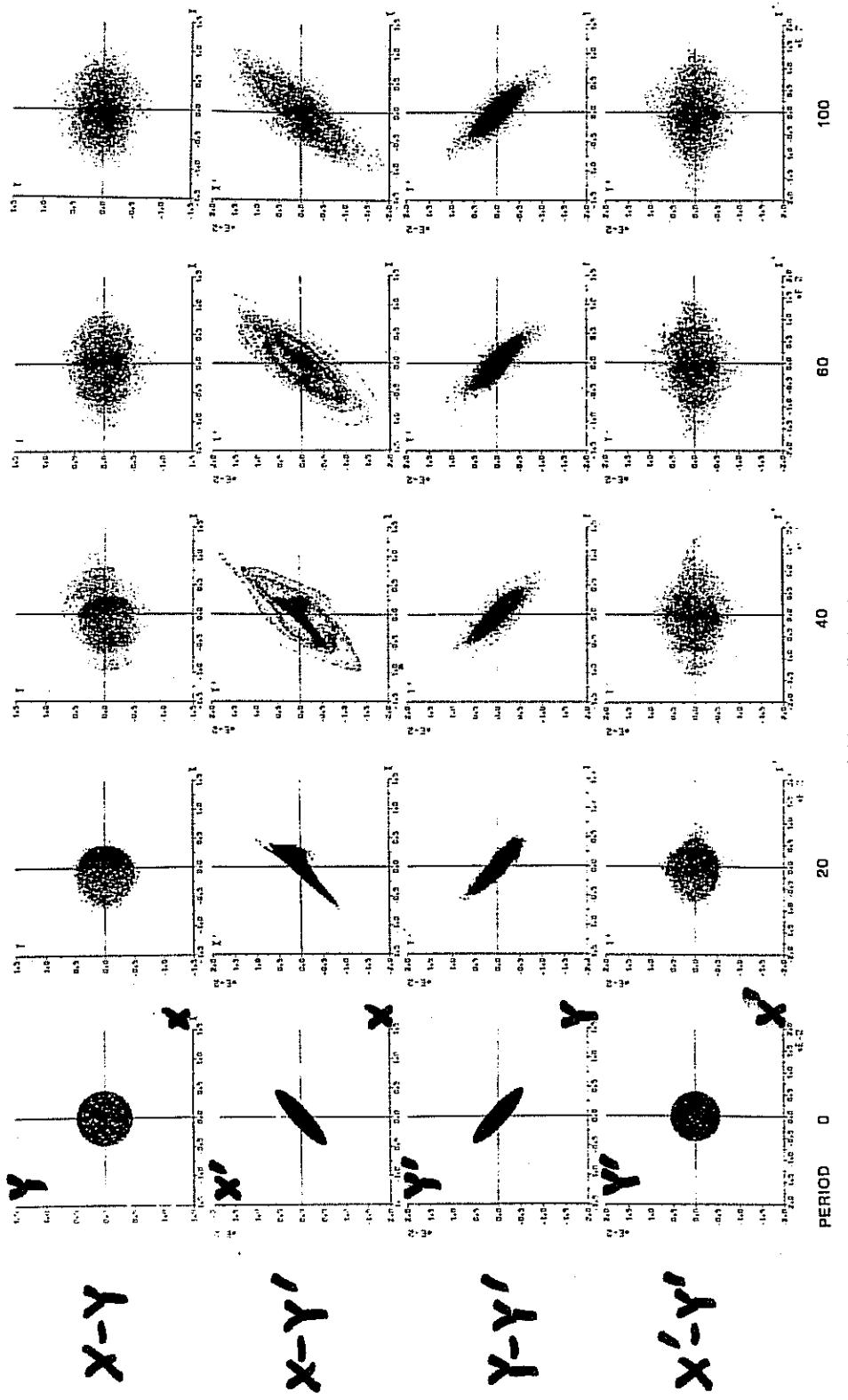


Lattice Periods

Lattice Periods



# Initial KV Distribution



See also Reiser Text, pg 495.

FIGURE 2 Transformation of an initial KV distribution through the GSI FODO channel at  $\sigma_0 = 90^\circ$ ,  $\sigma = 41^\circ$ .  
J. Struckmeier, J. Klabunde, and M. Reiser, Part. Accel., 15, 47 (1984).

# Initial Gaussian Distribution

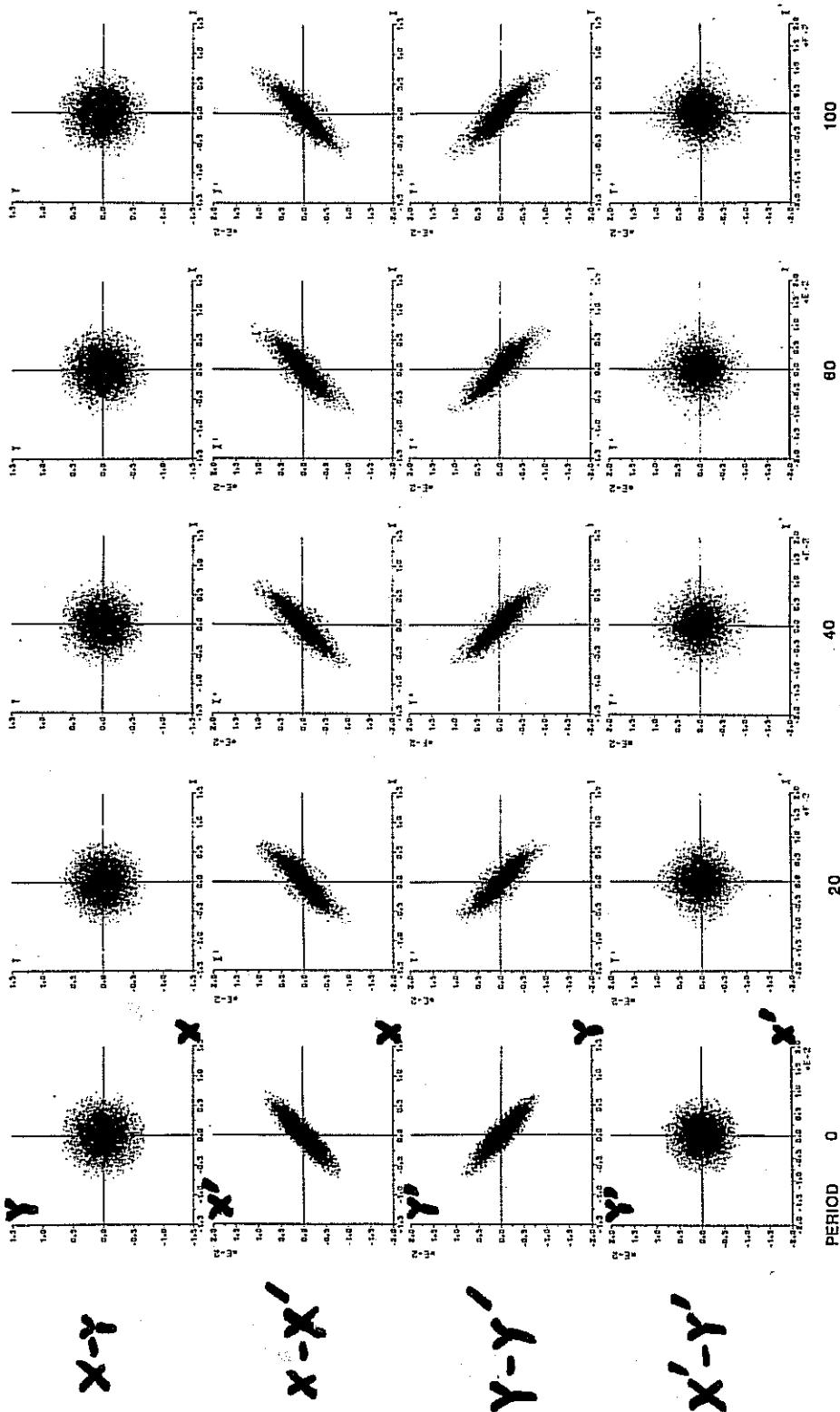


FIGURE 6 Transformation of an initial Gaussian distribution (rms-matched) through the GSI FODO channel at  $\sigma_0 = 90^\circ$ ,  $\sigma = 41^\circ$ .  
 J. Struckmeier, J. Klabunde and M. Reiser, Part. Accel., 15; 47 (1977)

## GROWTH OF DIFFERENT DISTRIBUTIONS

61

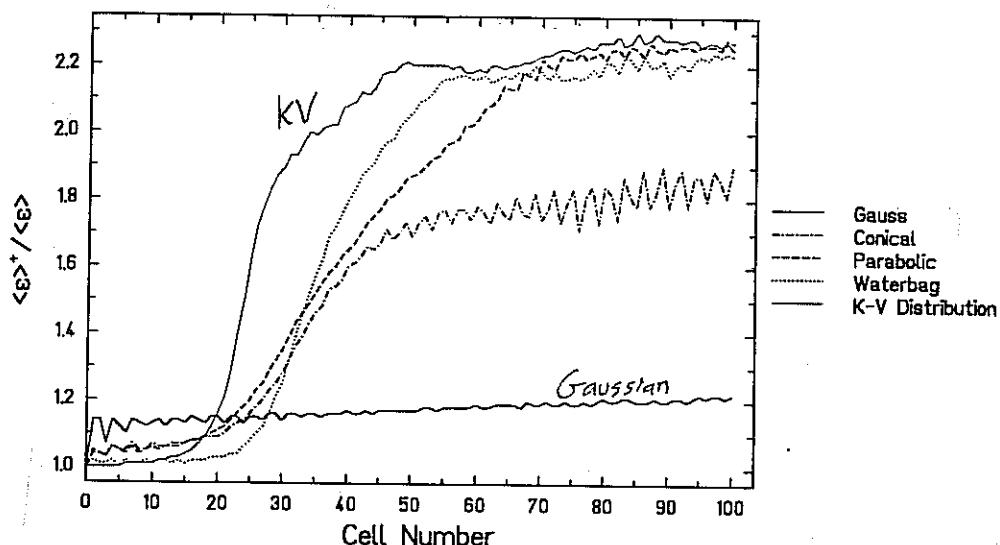
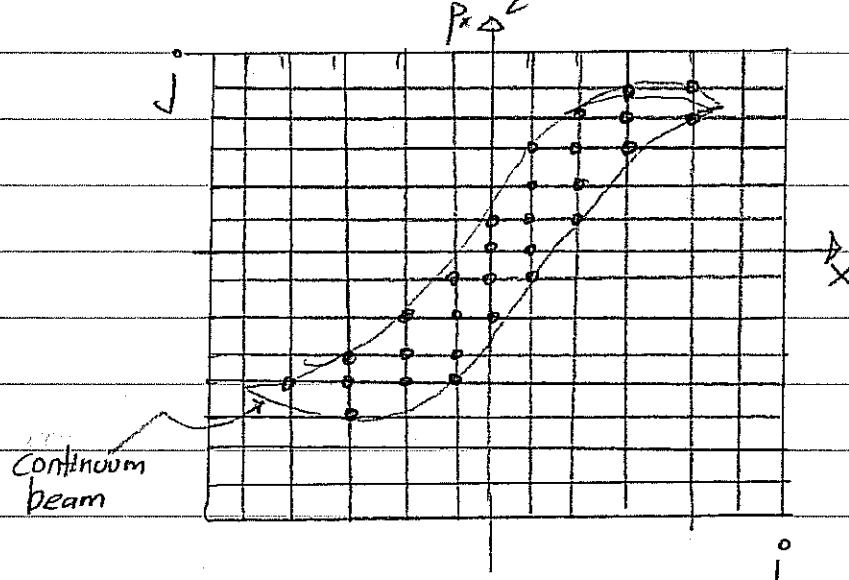


FIGURE 7 Emittance growth factors versus the number of cells obtained from particle simulations for initial K-V, waterbag, parabolic, conical and Gaussian distributions at  $\sigma_0 = 90^\circ$ ,  $\sigma = 41^\circ$ .

From J. Struckmeier, J. Kabunde, and M. Reiser,  
Part. Accel. 15 47 (1984).

## Distribution Methods - Direct Gridded Solution of Vlasov's Equation.

Consider the Vlasov equation as an example.



$$f(x_i, p_j, t) = f_{ij}(t)$$

The distribution is advanced at discrete grid points in time.

- Fields are typically solved using a discrete spatial mesh as for the particle methods described before.
  - Deposition on mesh is typically straightforward in Vlasov case (sum over momentum variables)
- The distribution advance cycle is different than for particle methods.
  - Numerical stability is key.
  - Characteristics and "semi-Lagrangian" methods can be employed.
  - Methods for solving for characteristics are familiar from dynamics/plasma physics.

# Direct Vlasov Methods

S. M. Lund 66/

## "Pros"

Reasons for Vlasov simulations:

- Low noise - only discretization effects without statistical effects
- Allows clear analysis of collective effects and tenuous distribution components.

## "Cons"

Reasons why Vlasov simulations are presently employed less than PIC:

- Extreme memory requirements for needed grid resolution in multi-dimensional phase space.
- Beams often have sharp edges in phase space that move in response to varying applied focusing forces.
- Numerical stability tends to be more difficult than in particle methods.

No time to illustrate Vlasov and other distribution methods in this introductory course. We hope to cover these methods and other numerical fieldsolve techniques in a later expanded version of this class.

However, ... it is easy to generalize from what we have learned!

## WARP Code Overview

Electrostatic Multi-dimensional PIC Code

WARP3d -  $x, y, z, p_x, p_y, p_z$   
Moves Int

Many Fieldsolvers!  
SOR, Multigrid, FFT,  
FFT + Tridiag, FFT + Cap Matrix, ...

WARPxy -  $x, y, p_x, p_y, p_z$   
Moves In s

WARPxz -  $r, z, p_r, p_z, p_\theta$   
Moves Int

WARPenv - envelope solver  
used to seed / load PIC  
 $f_x, f_y, f_x', f_y'$   
Advances In s

Hermes - Fluid II + Bridded  
Space Charge Field

- Common diagnostic tools built around g1st graphics
- Run with python interpreter

Example Script

"ag-slice.py"

Acknowledgements, ReferencesNumerical Methods

1. Forman S. Acton, "Numerical Methods that Work," Harper and Row Publishers, New York, 1970.

2. Stephen Koon, "Computational Physics," Addison-Wesley Publishing Co., 1986.

Survey of Numerical Methods

3. W. Press, B. Flannery, S. Teukolsky, W. Vetterling, "Numerical Recipes, The Art of Scientific Computing," Cambridge University Press, 1986.

Particle Methods

4. C. K. Birdsall and A. B. Langdon, "Plasma Physics via Computer Simulation," McGraw-Hill Book Company, 1985.

5. R.W. Hockney and J.W. Eastwood, "Computer Simulation Using Particles," Institute of Physics Publishing, 1988.

A special thanks to Alex Friedman, Dave Grote, Jean-Luc Vay, and Michiel de Hoon of the Lawrence Livermore and Lawrence Berkeley National Labs for help with these notes and general guidance. Also thanks are due for Irving Haber (NRL), Christine Celata (LBL), and Bill Fawley (LBL) for educating the authors on various simulation methods.